**THIẾT KẾ VÀ LẬP TRÌNH WEB**

Mục lục

[1. TỔNG QUAN WEBSITE 12](#_Toc23660521)

[1.1 Một số khái niệm 12](#_Toc23660522)

[1.2 Tổng quan các Công nghệ Web 12](#_Toc23660523)

[1.3 Giới thiệu cách thức hoạt động của website 12](#_Toc23660524)

[1.4 Một số kiến trúc dịch vụ web 12](#_Toc23660525)

[2 THIẾT KẾ WEBSITE 13](#_Toc23660526)

[2.1 HTML 13](#_Toc23660527)

[2.1.1 Giới thiệu về HTML 13](#_Toc23660528)

[2.1.2 Chỉnh sửa HTML 14](#_Toc23660529)

[2.1.3 Phần tử HTML 15](#_Toc23660530)

[2.1.4 Các thuộc tính HTML 16](#_Toc23660531)

[2.1.5 Các tiêu đề HTML (HTML Headings) 17](#_Toc23660532)

[2.1.6 HTML Paragraphs 18](#_Toc23660533)

[2.1.7 HTML Styles 19](#_Toc23660534)

[2.1.8 Định dạng văn bản HTML 19](#_Toc23660535)

[2.1.9 HTML Quotations 20](#_Toc23660536)

[2.1.10 HTML Comments 21](#_Toc23660537)

[2.1.11 HTML Colors 21](#_Toc23660538)

[2.1.12 HTML CSS 29](#_Toc23660539)

[2.1.13 HTML Links 32](#_Toc23660540)

[2.1.14 HTML5 35](#_Toc23660541)

[2.1.15 HTML Images 41](#_Toc23660542)

[2.1.16 HTML Tables 42](#_Toc23660543)

[2.1.17 HTML Lists 49](#_Toc23660544)

[2.1.18 HTML Blocks 54](#_Toc23660545)

[2.1.19 HTML Classes 56](#_Toc23660546)

[2.1.20 HTML Id 59](#_Toc23660547)

[2.1.21 HTML Forms 61](#_Toc23660548)

[2.1.22 HTML Examples 78](#_Toc23660549)

[2.1.23 HTML References 78](#_Toc23660550)

[2.1.24 HTML5 93](#_Toc23660551)

[2.1.25 HTML Graphics 112](#_Toc23660552)

[2.1.26 HTML Media 115](#_Toc23660553)

[2.1.27 HTML APIs 119](#_Toc23660554)

[2.1.28 HTML Examples 125](#_Toc23660555)

[2.1.29 HTML References 125](#_Toc23660556)

[2.2 JavaScript 141](#_Toc23660557)

[2.2.1 Căn bản về JavaScipt 141](#_Toc23660558)

[2.3 1.Biểu thức chính quy là gì? 187](#_Toc23660559)

[2.4 2.Cú pháp 187](#_Toc23660560)

[*Chức năng:* 207](#_Toc23660561)

[*Vòng lặp:* 207](#_Toc23660562)

[*Điều kiện:* 208](#_Toc23660563)

[2.4.2 2.2.2 Biểu mẫu và xác thực dữ liệu nhập 227](#_Toc23660564)

[2.4.3 2.2.3 Mô hình đối tượng tài liệu (DOM – Document Oject Model) 229](#_Toc23660565)

[2.4.4 229](#_Toc23660566)

[2.4.5 2.2.4 Mô hình đối tượng trình duyệt (BOM – Browser Object Model) 243](#_Toc23660567)

[2.4.6 2.2.5 Lập trình AJAX 249](#_Toc23660568)

[2.4.7 2.2.6 Sử dụng JSON 272](#_Toc23660569)

[**2.3 CSS** 295](#_Toc23660570)

[***2.3.1CSS Tutorial(CSS)*** 295](#_Toc23660571)

[**2.3.1.1. CSS Introduction** 295](#_Toc23660572)

[**2.3.1.2 Cú pháp CSS** 296](#_Toc23660573)

[p {   color: red;   /\* This is a single-line comment \*/   text-align: center; } /\* This is a multi-line comment \*/ 297](#_Toc23660574)

[**2.3.1.3. Bộ chọn CSS** 297](#_Toc23660575)

[h1, h2, p {   text-align: center;   color: red; } 299](#_Toc23660576)

[**2.3.1.4 Cách thêm CSS** 299](#_Toc23660577)

[**2.3.1.5 MÀU CSS** 303](#_Toc23660578)

[**2.3.1.6 NỀN CSS** 305](#_Toc23660579)

[**2.3.1.7 BIÊN GIỚI CSS** 307](#_Toc23660580)

[**2.3.1.8 Margin CSS** 312](#_Toc23660581)

[**2.3.1.9 ĐỆM CSS** 315](#_Toc23660582)

[***2.3.1.10 CHIỀU CAO VÀ CHIỀU RỘNG CSS*** 317](#_Toc23660583)

[2.5 318](#_Toc23660584)

[2.5.1 2.3.1.11 MÔ HÌNH HỘP CSS 318](#_Toc23660585)

[2.6 Mô hình hộp CSS 318](#_Toc23660586)

[2.7 Tất cả các yếu tố HTML có thể được coi là hộp. Trong CSS, thuật ngữ "mô hình hộp" được sử dụng khi nói về thiết kế và bố cục.Mô hình hộp CSS về cơ bản là một hộp bao quanh mọi phần tử HTML. Nó bao gồm: lề, viền, phần đệm và nội dung thực tế. Hình ảnh dưới đây minh họa mô hình hộp: 318](#_Toc23660587)

[2.8 Giải thích về các phần khác nhau: 319](#_Toc23660588)

[2.9 · Nội dung - Nội dung của hộp, nơi xuất hiện văn bản và hình ảnh 319](#_Toc23660589)

[2.10 · Đệm - Xóa một khu vực xung quanh nội dung. Phần đệm trong suốt 319](#_Toc23660590)

[2.11 · Đường viền - Đường viền bao quanh phần đệm và nội dung 319](#_Toc23660591)

[2.12 · Ký quỹ - Xóa một khu vực bên ngoài biên giới. Lề là trong suốt 319](#_Toc23660592)

[2.13 Mô hình hộp cho phép chúng ta thêm một đường viền xung quanh các phần tử và để xác định khoảng trắng giữa các phần tử. 319](#_Toc23660593)

[2.14 Chiều rộng và chiều cao của một phần tử 319](#_Toc23660594)

[2.15 Để đặt chính xác chiều rộng và chiều cao của một phần tử trong tất cả các trình duyệt, bạn cần biết mô hình hộp hoạt động như thế nào. 319](#_Toc23660595)

[2.16 Ví dụ: 319](#_Toc23660596)

[2.16.1 2.3.1.12 PHÁC THẢO CSS 319](#_Toc23660597)

[2.17 Phác thảo CSS 319](#_Toc23660598)

[2.18 Một phác thảo là một đường được vẽ xung quanh các phần tử, BÊN NGOÀI các đường viền, để làm cho phần tử "nổi bật" 319](#_Toc23660599)

[2.19 CSS có các thuộc tính phác thảo sau: 319](#_Toc23660600)

[2.20 · outline-style 319](#_Toc23660601)

[2.21 · outline-color 319](#_Toc23660602)

[2.22 · outline-width 319](#_Toc23660603)

[2.23 · outline-offset 319](#_Toc23660604)

[2.24 · outline 320](#_Toc23660605)

[2.25 Phong cách phác thảo CSS 320](#_Toc23660606)

[2.26 Các outline-stylebất động sản xác định phong cách của đề cương, và có thể có một trong các giá trị sau: 320](#_Toc23660607)

[2.27 - dotted - Xác định một phác thảo chấm 320](#_Toc23660608)

[2.28 - dashed - Xác định một phác thảo nét 320](#_Toc23660609)

[2.29 - solid - Xác định một phác thảo vững chắc 320](#_Toc23660610)

[2.30 - double - Xác định một phác thảo kép 320](#_Toc23660611)

[2.31 - groove - Xác định đường viền có rãnh 3D 320](#_Toc23660612)

[2.32 - ridge - Xác định phác thảo có viền 3D 320](#_Toc23660613)

[2.33 - inset - Xác định phác thảo chèn 3D 320](#_Toc23660614)

[2.34 - - outset - Xác định phác thảo đầu ra 3D 320](#_Toc23660615)

[2.35 - none - Xác định không có phác thảo 320](#_Toc23660616)

[2.36 - hidden - Xác định một phác thảo ẩn 320](#_Toc23660617)

[2.37 Màu phác thảo CSS 320](#_Toc23660618)

[2.38 Các outline-colortài sản được sử dụng để thiết lập màu sắc của phác thảo. 320](#_Toc23660619)

[2.39 Màu sắc có thể được thiết lập bởi: 320](#_Toc23660620)

[2.40 - Tên - chỉ định tên màu, như "đỏ" 320](#_Toc23660621)

[2.41 - RGB - chỉ định giá trị RGB, như "rgb (255,0,0)" 320](#_Toc23660622)

[2.42 - Hex - chỉ định giá trị hex, như "# ff0000" 320](#_Toc23660623)

[2.43 - đảo ngược - thực hiện đảo ngược màu (đảm bảo có thể nhìn thấy đường viền, bất kể nền màu) 320](#_Toc23660624)

[2.44 Chiều rộng phác thảo CSS 320](#_Toc23660625)

[2.45 Các outline-widthtài sản quy định cụ thể rộng đường viền, và có thể có một trong các giá trị sau: 321](#_Toc23660626)

[2.46 - mỏng (thường là 1px) 321](#_Toc23660627)

[2.47 - trung bình (thường là 3px) 321](#_Toc23660628)

[2.48 - dày (thường là 5px) 321](#_Toc23660629)

[2.49 - Một kích thước cụ thể (tính bằng px, pt, cm, em, v.v.) 321](#_Toc23660630)

[2.50 5CSS Outline - Thuộc tính tốc ký 321](#_Toc23660631)

[2.51 Các outlinebất động sản là một tài sản viết tắt cho thiết lập các thuộc tính outline cá nhân sau đây: 321](#_Toc23660632)

[2.52 - outline-width 321](#_Toc23660633)

[2.53 - outline-style (cần thiết) 321](#_Toc23660634)

[2.54 - outline-color 321](#_Toc23660635)

[2.55 Các outlinetài sản được quy định như một, hai, hoặc ba giá trị từ danh sách trên. Thứ tự của các giá trị không quan trọng. 321](#_Toc23660636)

[2.56 Bù đắp phác thảo CSS 321](#_Toc23660637)

[2.57 Các outline-offsetbất động sản cho biết thêm khoảng trống giữa một phác thảo và cạnh / biên giới của một phần tử. Không gian giữa một yếu tố và phác thảo của nó là trong suốt. 321](#_Toc23660638)

[2.58 Tất cả các thuộc tính phác thảo CSS 321](#_Toc23660639)

[2.59 321](#_Toc23660640)

[**2.3.1.13 CSS TIÊU ĐỀ** 321](#_Toc23660641)

[**2.3.1.14 CSS Fonts** 325](#_Toc23660642)

[**2.3.1.15 CSS icons** 329](#_Toc23660643)

[**2.3.1.16 CSS links** 330](#_Toc23660644)

[***2.3.1.17 CSS lits*** 334](#_Toc23660645)

[2.60 Thuộc tính list-style-type trong CSS 335](#_Toc23660646)

[2.61 Thuộc tính list-style-position trong CSS 336](#_Toc23660647)

[2.62 Thuộc tính list-style-image trong CSS 336](#_Toc23660648)

[2.63 Thuộc tính list-style trong CSS 337](#_Toc23660649)

[2.64 Thuộc tính marker-offset trong CSS 337](#_Toc23660650)

[2.65 Tạo màu cho List trong CSS 337](#_Toc23660651)

[***2.3.1.18 CSS tables*** 337](#_Toc23660652)

[***2.3.1.19 CSS displays*** 338](#_Toc23660653)

[***2.3.1.20 CSS max-width*** 339](#_Toc23660654)

[***2.3.1.21 CSS Position*** 340](#_Toc23660655)

[***2.3.1.22 CSS overflow*** 341](#_Toc23660656)

[***2.3.1.23 CSS float*** 341](#_Toc23660657)

[***2.3.1.24 CSS inline-block*** 342](#_Toc23660658)

[***2.3.1.25 CSS align*** 342](#_Toc23660659)

[***2.3.1.26 CSS combinators*** 343](#_Toc23660660)

[2.66 Child combinator 344](#_Toc23660661)

[2.67 Adjacent sibling combinator 345](#_Toc23660662)

[2.68 General sibling combinator 345](#_Toc23660663)

[***2.3.1.27 CSS Pseudo-class*** 346](#_Toc23660664)

[***2.3.1.28 CSS Pseudo-element*** 352](#_Toc23660665)

[***2.3.1.29 CSS Opacity*** 355](#_Toc23660666)

[***2.3.1.30 CSS Navigation Bar*** 357](#_Toc23660667)

[***2.3.1.31 CSS Dropdowns*** 362](#_Toc23660668)

[***2.3.1.32 CSS Image Gallery*** 365](#_Toc23660669)

[***2.3.1.33CSS Image Sprites*** 366](#_Toc23660670)

[***2.3.1.34 CSS Attr Selectors*** 368](#_Toc23660671)

[***2.3.1.35 CSS Forms*** 370](#_Toc23660672)

[***2.3.1.36 CSS Counters*** 373](#_Toc23660673)

[***2.3.1.37 CSS Website Layout*** 375](#_Toc23660674)

[***2.3.1.38 CSS Units*** 378](#_Toc23660675)

[***2.3.1.39 CSS Specificity*** 380](#_Toc23660676)

[**2.3.2 CSS Advanced** 381](#_Toc23660677)

[***2.3.2.1 CSS Rounded Corners*** 381](#_Toc23660678)

[*Góc làm tròn CSS* 381](#_Toc23660679)

[***2.3.2.2 CSS Border images*** 384](#_Toc23660680)

[***2.3.2.3 CSS Colors*** 385](#_Toc23660681)

[***2.3.2.4. CSS Gradients*** 389](#_Toc23660682)

[***2.3.2.6 CSS Shadows*** 393](#_Toc23660683)

[***2.3.2.8 CSS Web Fonts*** 395](#_Toc23660684)

[**2.3.2.9 CSS 2D Transforms** 399](#_Toc23660685)

[**2.3.2.10 CSS 3D Transforms** 400](#_Toc23660686)

[**2.3.2.11 CSS Transitions** 402](#_Toc23660687)

[**2.3.2.12 CSS Animations** 404](#_Toc23660688)

[**2.3.2.13 CSS tooltips** 407](#_Toc23660689)

[**2.3.2.14 CSS Style Images** 409](#_Toc23660690)

[***2.3.2.15 CSS Object-fit*** 413](#_Toc23660691)

[***2.3.2.16 CSS Buttons*** 413](#_Toc23660692)

[***2.3.2.18 CSS Multiple Columns*** 418](#_Toc23660693)

[***2.3.2.19 CSS User Interface*** 420](#_Toc23660694)

[***2.3.2.20 CSS Variables*** 422](#_Toc23660695)

[***2.3.2.21 CSS Box sizing*** 423](#_Toc23660696)

[***2.3.2.22 CSS Flexbox*** 425](#_Toc23660697)

[***2.3.2.23 CSS Media Queries*** 433](#_Toc23660698)

[***2.3.2.24 CSS MQ Examples*** 435](#_Toc23660699)

[**2.3.3 Responsive** 438](#_Toc23660700)

[***2.3.3.1. RWD INTRO*** 438](#_Toc23660701)

[***2.3.3.2. RWD viewport*** 439](#_Toc23660702)

[***2.3.3.3. RWD Grid view*** 441](#_Toc23660703)

[***2.3.3.4 RWD Media queries*** 442](#_Toc23660704)

[***2.3.3.5 RWD Images*** 445](#_Toc23660705)

[***2.3.3.6 RWD Videos*** 448](#_Toc23660706)

[***2.3.3.7 RWD Framework*** 448](#_Toc23660707)

[**2.3.3.8 RWD Templates** 450](#_Toc23660708)

[**2.3.4 Gird** 452](#_Toc23660709)

[***2.3.4.2 Grid Container*** 454](#_Toc23660710)

[***2.3.4.3 Grid Item*** 454](#_Toc23660711)

[**2.3.5 References** 458](#_Toc23660712)

[**2.3.5.1. CSS References** 458](#_Toc23660713)

[2.69 458](#_Toc23660714)

[2.70 460](#_Toc23660715)

[2.71 466](#_Toc23660716)

[2.72 468](#_Toc23660717)

[2.73 468](#_Toc23660718)

[2.74 468](#_Toc23660719)

[2.75 472](#_Toc23660720)

[2.76 474](#_Toc23660721)

[2.77 474](#_Toc23660722)

[2.78 475](#_Toc23660723)

[2.79 475](#_Toc23660724)

[2.80 475](#_Toc23660725)

[2.81 476](#_Toc23660726)

[2.82 477](#_Toc23660727)

[2.83 479](#_Toc23660728)

[2.84 480](#_Toc23660729)

[2.85 480](#_Toc23660730)

[2.86 481](#_Toc23660731)

[2.87 481](#_Toc23660732)

[2.88 484](#_Toc23660733)

[2.89 484](#_Toc23660734)

[2.90 484](#_Toc23660735)

[2.91 485](#_Toc23660736)

[**2.3.5.4. Chức năng CSS** 485](#_Toc23660737)

[**2.3.5.6 CSS Animatable** 488](#_Toc23660738)

[**2.3.5.7 CSS Units** 491](#_Toc23660739)

[***2.3.5.8 CSS PX-EM Converter*** 493](#_Toc23660740)

[***2.3.5.9 CSS Colors*** 493](#_Toc23660741)

[***2.3.5.10 CSS Colors Values*** 499](#_Toc23660742)

[***2.3.5.11 CSS Default Values*** 501](#_Toc23660743)

[***2.3.5.12 CSS Entities*** 514](#_Toc23660744)

[2.92 Bộ chọn CSS 734](#_Toc23660745)

DANH MỤC BẢNG BIỂU

Bảng 1.1. Một số yếu tố ảnh hưởng đến thành công dự án

Bảng 1.2

DANH MỤC HÌNH VẼ

**DANH MỤC CHỮ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| **STT** | **Chữ cái viết tắt/ kí hiệu** | **Cụm từ đầy đủ** |
| 1 | CNTT | Công nghệ thông tin |
| 2 | IT | Viết tắt của từ Information Technology – Công nghệ thông tin |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# TỔNG QUAN WEBSITE

## Một số khái niệm

## Tổng quan các Công nghệ Web

## Giới thiệu cách thức hoạt động của website

## Một số kiến trúc dịch vụ web

# THIẾT KẾ WEBSITE

*Chương này giới thiệu lập các bước để thiết kế website bao gồm thiết kế kiến trúc, thiết kế giao diện, thiết kế cơ sở dư liệu. Giới thiệu các bước để triển khai website.*

1. Thiết kế website (Thảo)
   1. Thiết kế kiến trúc.
   2. Thiết kế giao diện.
   3. Thiết kế cơ sở dữ liệu.
2. Triển khai website (Thảo)
   1. Các bước triển khai website.
   2. Cài đặt và kiểm tra.
3. Cấu trúc của một trang HTML (Mai) (HTML)

TML

## HTML

### Giới thiệu về HTML

* HTML là gì?

HTML là ngôn ngữ đánh dấu chuẩn để tạo ra các trang Web. HTML là viết tắt của Hyper Text Markup Language. HTML mô tả cấu trúc của các trang Web sử dụng đánh dấu. Các phần tử HTML được biểu diễn bằng các thẻ. Thẻ HTML gắn nhãn các nội dung như: “tiêu đề”, “đoạn”, “bảng”,…. Trình duyệt không hiển thị các thẻ HTML nhưng sử dụng chúng để hiển thị nội dung của trang

* Ví dụ về một tài liệu HTML đơn giản:

<!DOCTYPE html>  
<html>  
<head>  
<title>Page Title</title>  
</head>  
<body>  
<h1>My First Heading</h1>  
<p>My first paragraph.</p>  
</body>  
</html>

Khai báo <!DOCTYPE html> xác định tài liệu này là HTML5

Phần tử <html> là phần tử gốc của một trang HTML

Phần tử <head> chứa thông tin meta về tài liệu

Phần tử <title> định nghĩa tiêu đề cho tài liệu

Phần tử <body> chứa nội dung trang hiển thị

Phần tử <h1> định nghĩa một tiêu đề lớn

Phần tử <p> định nghĩa một đoạn

* Thẻ HTML

Thẻ HTML là các tên phần tử được bao quanh bởi các dấu ngoặc nhọn:

Cú pháp:

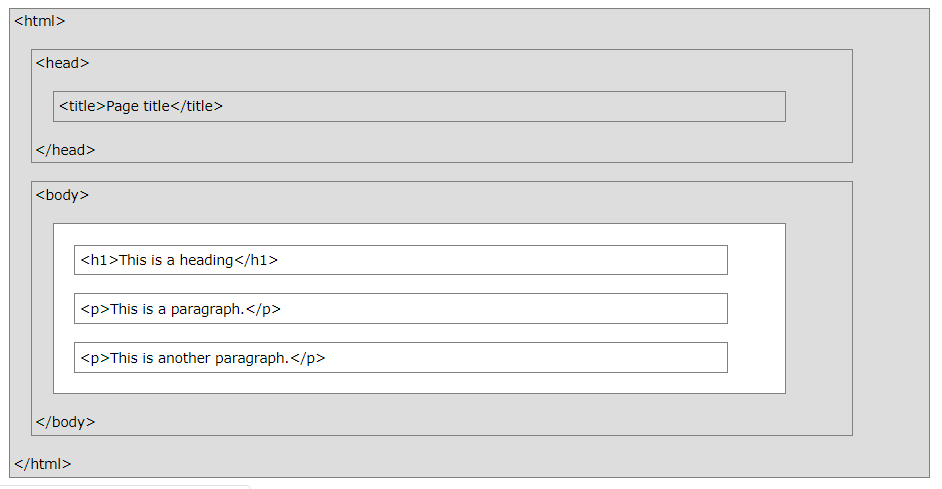
<tên thẻ> nội dung </ tên thẻ>

Các thẻ HTML thường đi theo cặp như <p> và </ p>. Thẻ đầu tiên trong một cặp là thẻ bắt đầu, thẻ thứ hai là thẻ kết thúc. Thẻ kết thúc được viết giống như thẻ bắt đầu, nhưng có dấu gạch chéo được chèn trước tên thẻ. Thẻ bắt đầu còn được gọi là thẻ mở và thẻ kết thúc là thẻ đóng.

* Trình duyệt web

Nhiệm vụ của trình duyệt web (Chrome, IE, Firefox, Safari) là đọc tài liệu HTML và hiển thị chúng. Trình duyệt không hiển thị các thẻ HTML, nhưng sử dụng chúng để xác định cách hiển thị tài liệu

* Cấu trúc trang HTML



Lưu ý: Chỉ nội dung bên trong phần <body> mới được hiển thị trong trình duyệt.

* Khai báo <! DOCTYPE>

Khai báo <! DOCTYPE> đại diện cho loại tài liệu và giúp các trình duyệt hiển thị các trang web một cách chính xác. Nó chỉ phải xuất hiện một lần, ở đầu trang (trước bất kỳ thẻ HTML nào). Khai báo này không phân biệt chữ hoa chữ thường. Khai bào <! DOCTYPE> cho HTML5 là: <! DOCTYPE html>

Phiên bản HTML: Đã có nhiều phiên bản HTML như

|  |  |
| --- | --- |
| Phiên bản | Năm |
| HTML | 1991 |
| HTML 2.0 | 1995 |
| HTML 3.2 | 1997 |
| HTML 4.01 | 1999 |

### Chỉnh sửa HTML

Các trang web có thể được tạo và sửa đổi bằng cách sử dụng các trình soạn thảo HTML chuyên nghiệp. Tuy nhiên, đối với việc học HTML có thể sử dụng một trình soạn thảo văn bản đơn giản như Notepad (PC) hoặc TextEdit (Mac). Đây là một cách để để soạn thảo HTML hay và đơn giản nhất. Chúng ta thực hiện bốn bước dưới đây để tạo trang web đầu tiên của bạn bằng Notepad hoặc TextEdit.

Bước 1: Mở Notepad (PC) Window 8 trở lên:

Đối với hệ điều hành Window 8 trở lên: Mở màn hình khởi động (biểu tượng cửa sổ ở dưới cùng bên trái trên màn hình của bạn). Nhập Notepad.

Đối với hệ điều hành Windows 7 trở về trước:

Start > Program> Accessorries> Notepad

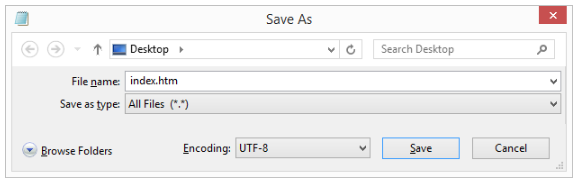
Bước 2: Viết HTML

Viết hoặc gõ HTML vào Notepad.

<!DOCTYPE html>  
<html>  
<body>  
<h1>My First Heading</h1>  
<p>My first paragraph.</p>  
</body>  
</html>

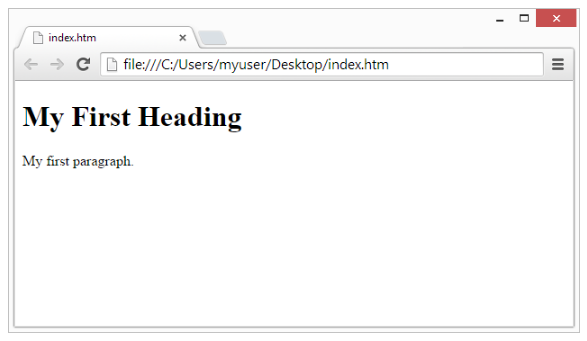
Bước 3: Lưu trang HTML

Lưu tệp trên máy tính của bạn. Chọn File > Save as trong menu Notepad.

Đặt tên tệp là "index.htm" và đặt mã hóa thành UTF-8 (là mã hóa ưu tiên cho các tệp HTML).

Bước 4: Xem trang HTML trong trình duyệt

Mở tệp HTML đã lưu trong trình duyệt yêu thích của bạn (nhấp đúp vào tệp hoặc nhấp chuột phải - và chọn "Open with"). Kết quả sẽ được hiển thị:



### Phần tử HTML

* Phần tử HTML:

Một phần tử HTML thường bao gồm thẻ bắt đầu và thẻ kết thúc, với nội dung được chèn vào giữa: <tên thẻ> Nội dung ... </ tên thẻ>. Phần tử HTML là tất cả những thứ từ thẻ mở đến thẻ đóng:

|  |  |  |
| --- | --- | --- |
| Thẻ mở | Nội dung | Thẻ đóng |
| <h1> | Tiêu đề | </ h1> |
| <p> | Đoạn | </ p> |
| <br> |  |  |

Chú ý: Các phần tử HTML không có nội dung được gọi là các phần tử rỗng. Các phần tử trống không có thẻ kết thúc, chẳng hạn như phần tử <br> (cho biết dấu ngắt dòng).

* Các phần tử HTML lồng nhau

Các phần tử HTML có thể được lồng vào nhau (các phần tử có thể chứa các phần tử). Tất cả các tài liệu HTML bao gồm các phần tử HTML lồng nhau. Ví dụ này chứa bốn phần tử HTML:

<!DOCTYPE html>  
<html>  
<body>  
  
<h1>My First Heading</h1>

<p>My first paragraph.</p>

</body>  
</html>

Giải thích:

Phần tử <html> định nghĩa toàn bộ tài liệu. Nó có thẻ mở <html> và thẻ đóng </ html>. Nội dung phần tử là một phần tử HTML khác (phần tử <body>)

Phần tử <body> xác định phần thân của tài liệu. Nó có thẻ đóng <body> và thẻ đóng </ body>. Nội dung phần tử là hai phần tử HTML khác (<h1> và <p>).

Phần tử <h1> định nghĩa một tiêu đề. Nó có thẻ mở <h1> và thẻ đóng </ h1>. Nội dung phần tử là: “My First Heading”

Phần tử <p> định nghĩa một đoạn văn. Nó có thẻ mở là <p> và thẻ đóng </p>. Nội dung phần tử là “My First paragraph”

Chú ý: Đừng nên quên thẻ đóng mặc dù trong một số trường hợp các phần tử HTML vẫn hiển thị đúng cho dù ta quên sử dụng thẻ đóng

* Các phần tử HTML rỗng

Các phần tử HTML không có nội dung được gọi là các phần tử rỗng. <br> là phần tử trống không có thẻ đóng (thẻ <br> định nghĩa cho việc ngắt dòng). Các phần tử trống có thể được "đóng" trong thẻ mở như sau: <br />.

HTML5 không yêu cầu phải đóng các phần tử rỗng. Nhưng nếu muốn xác nhận chặt chẽ hơn, hoặc nếu cần làm cho tài liệu của bạn có thể đọc được bằng các trình phân tích cú pháp XML, bạn phải đóng tất cả các phần tử HTML đúng cách.

### Các thuộc tính HTML

Tất cả các phần tử HTML có thể có các thuộc tính. Các thuộc tính cung cấp thông tin bổ sung về một phần tử. Các thuộc tính luôn được đặt trong thẻ bắt mở. Thuộc tính thường đặt trong tên hoặc các cặp giá trị như: tên = "giá trị"

* Thuộc tính “href”

Các liên kết HTML được định nghĩa với thẻ <a>. Địa chỉ liên kết được chỉ định trong thuộc tính href:

Ví dụ: <a href="https://www.neu.edu.vn">This is a link </a>

* Thuộc tính “scr”

Các hình ảnh HTML được định nghĩa bằng thẻ <img>.Tên tệp của nguồn hình ảnh được chỉ định trong thuộc tính src:

Ví dụ: <img src="flower.jpg">

* Thuộc tính chiều rộng và chiều cao

Những hình ảnh trong HTML có một tập hợp các thuộc tính kích thước xác định chiều rộng và chiều cao của hình ảnh:

Ví dụ: <img src="flower.jpg" width="500" height="600">

Kích thước hình ảnh được chỉ định bằng pixel: width = "500" có nghĩa là rộng 500 pixel và height = "600" có nghĩa là cao 600 pixel

* Thuộc tính “alt”

Thuộc tính alt chỉ định một văn bản thay thế được sử dụng, khi một hình ảnh không thể được hiển thị. Giá trị của thuộc tính có thể được đọc bởi trình đọc màn hình. Bằng cách này, ai đó "đang nghe" trang web, ví dụ: một người khiếm thị, có thể "nghe" phần tử đó.

Ví dụ: <img src="flower.jpg" alt "Bông hoa">

* . Thuộc tính “slyle”

Thuộc tính “style” được sử dụng để xác định kiểu dáng của một phần tử, như màu, phông chữ, kích thước, v.v.

Ví dụ <p style="color:red">Đây là một đoạn </p>

* . Thuộc tính “lang”

Ngôn ngữ của tài liệu có thể được khai báo trong thẻ <html>. Ngôn ngữ được khai báo với thuộc tính “lang”. Khai báo một ngôn ngữ là quan trọng đối với các ứng dụng hỗ trợ (trình đọc màn hình) và các công cụ tìm kiếm:

Ví dụ:

<!DOCTYPE html>  
<html lang="en-US">  
<body>  
  
...  
  
</body>  
</html>

Hai chữ cái đầu tiên trong thuộc tính “lang” chỉ định ngôn ngữ (en). Nếu đó là tiếng địa phương/thổ ngữ thì sử dụng thêm hai chữ cái nữa (US).

* . Thuộc tính “title”

Ở đây, một thuộc tính “title” được thêm vào phần tử <p>. Giá trị của thuộc tính này sẽ được hiển thị dưới dạng chú thích khi di chuột qua

<p title="I'm a tooltip">  
This is a paragraph.  
</p>

Một số lưu ý:

* HTML5 không yêu cầu tên thuộc tính phải viết chữ thường. Tức là: thuộc tính “title” có thể được viết bằng chữ hoa hoặc chữ thường như “title” hoặc “TITLE”. Trong tài liệu này chúng tôi đề xuất viết chữ thường trong HTML và yêu cầu chữ thường cho các loại tài liệu chặt chẽ hơn như XHTML.
* HTML5 không yêu cầu dấu ngoặc kép xung quanh các giá trị thuộc tính. Chẳng hạn: Thuộc tính href, được minh họa ở trên, có thể được viết mà không có dấu ngoặc kép như sau:

<a href= <https://www.neu.edu.vn>>

* Các dấu nháy kép xung quanh các giá trị thuộc tính là phổ biến nhất trong HTML, nhưng cũng có thể sử dụng các dấu nháy đơn. Trong một số trường hợp, khi bản thân giá trị thuộc tính chứa dấu ngoặc kép, cần sử dụng dấu nháy đơn:

<p title='John "ShotGun" Nelson'>

Hoặc ngược lại:

<p title="John 'ShotGun' Nelson">

### Các tiêu đề HTML (HTML Headings)

Các tiêu đề được xác định bằng các thẻ từ <h1> đến <h6>. Trong đó: <h1> xác định tiêu đề quan trọng nhất. Và <h6> xác định tiêu đề ít quan trọng nhất. Trình duyệt tự động thêm một số khoảng trắng (lề) trước và sau tiêu đề. Công cụ tìm kiếm sử dụng các tiêu đề để lập chỉ mục cấu trúc và nội dung của các trang web. Người dùng lướt qua các trang theo các tiêu đề của nó. Điều quan trọng là sử dụng các tiêu đề để hiển thị cấu trúc tài liệu. Tiêu đề <h1> được sử dụng cho các tiêu đề chính, tiếp theo là tiêu đề <h2>, sau đó <h3> ít quan trọng hơn, v.v. Lưu ý: Chỉ sử dụng tiêu đề HTML cho tiêu đề mà không sử dụng tiêu đề để tạo các chữ lớn hoặc in đậm.

* Thay đổi kích thước tiêu đề

Mỗi tiêu đề HTML đều có kích thước mặc định. Tuy nhiên, chúng ta có thể chỉ định kích thước cho bất kỳ tiêu đề nào có thuộc tính style, sử dụng thuộc tính kích thước phông chữ CSS. Ví dụ: chỉ định cho tiêu đề 1 có kích thước 60px như sau:

<h1 style="font-size:60px;">Heading 1</h1>

Thẻ <hr> xác định ngắt theo chủ đề trong trang HTML và thường được hiển thị dưới dạng quy tắc ngang. Phần tử <hr> được sử dụng để tách nội dung hoặc xác định một sự thay đổi trong một trang HTML

* Quy tắc HTML Horizontal

Thẻ <hr> xác định ngắt theo chủ đề trong trang HTML và thường được hiển thị dưới dạng quy tắc ngang. Phần tử <hr> được sử dụng để tách nội dung hoặc xác định một sự thay đổi trong một trang HTML

* Phần tử The HTML <head> Element

Phần tử HTML <head> không liên quan gì đến các tiêu đề HTML. Phần tử <head> là một vùng chứa cho metadata. Siêu dữ liệu HTML là dữ liệu về tài liệu HTML. Siêu dữ liệu không được hiển thị. (The <head> element is a container for metadata. HTML metadata is data about the HTML document. Metadata is not displayed.). Phần tử <head> được đặt giữa thẻ <html> và thẻ <body>

<!DOCTYPE html>  
<html>  
  
<head>  
  <title>My First HTML</title>  
  <meta charset="UTF-8">  
</head>  
  
<body>  
.  
.  
.

* Cách xem nguồn HTML?

Để xem mã nguồng HTML: Nhấp chuột phải vào trang HTML và chọn "View Page Source" (trong Chrome hoặc trong IE hoặc tương tự trong các trình duyệt khác. Thao tác này sẽ mở một cửa sổ chứa mã nguồn HTML của trang. Để kiểm tra một phần tử HTML: Nhấp chuột phải vào một phần tử (hoặc một vùng trống) và chọn "Inspect" hoặc "Inspect Element" để xem những yếu tố nào được tạo thành chúng ta sẽ thấy cả HTML và CSS.

### HTML Paragraphs

The p dùng để xuống dòng. Ví dụ:

<p>This is a paragraph.</p>  
<p>This is another paragraph.</p>

Lưu ý: Trình duyệt tự động thêm một số khoảng trắng (lề) trước và sau một đoạn.

Hiển thị HTML

Bạn không thể chắc chắn cách HTML sẽ được hiển thị. Màn hình lớn hoặc nhỏ, và các cửa sổ thay đổi kích thước sẽ tạo ra kết quả khác nhau. Với HTML, bạn không thể thay đổi đầu ra bằng cách thêm khoảng trắng hoặc dòng bổ sung trong mã HTML của mình. Trình duyệt sẽ xóa mọi khoảng trắng thừa và các dòng thừa khi trang được hiển thị:

Khi sử dụng chúng ta cũng đừng quên thẻ đóng

Hầu hết các trình duyệt sẽ hiển thị HTML chính xác ngay cả khi bạn quên thẻ kết thúc. Ví dụ như:

Thí dụ

<p>This is a paragraph.</p>  
<p>This is another paragraph.</p>

Ví dụ này sẽ hoạt động trong hầu hết các trình duyệt, nhưng đôi khi nó có thể tạo ra các kết quả hoặc lỗi không mong muốn.

Ngắt dòng

Phần tử HTML <br> xác định ngắt dòng. Có thể sử dụng thẻ <br> nếu bạn muốn ngắt dòng (một dòng mới) mà không bắt đầu một đoạn mới. Ví dụ:

<p>This is<br>a paragraph<br>with line breaks.</p>

Thẻ <br> là một thẻ trống, có nghĩa là nó không có thẻ kết thúc.

### HTML Styles

Thuộc tính HTML Style :

Cài đặt kiểu của thành phần HTML có thể được thực hiện bằng thuộc tính style. Thuộc tính style HTML có cú pháp sau:

<tagname style="*property*:*value;*">

Trong đó

The property is a CSS property. The value is a CSS value.

(Thuộc tính là một thuộc tính CSS. Giá trị là một giá trị CSS). Tìm hiểu về CSS sẽ được mô tả ở phần tiếp theo trong cuốn bài giảng này

Màu nền HTML:

Thuộc tính "background-color” xác định màu nền cho thành phần HTML. Ví dụ, nếu muốn đặt màu nền cho một trang thành Powderblue có thể sử dụng câu lệnh sau:

<body style="background-color:powderblue;">  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>  
</body>

Màu chữ HTML

Thuộc tính "color” xác đinh màu chữ cho thành phần HTML. Chẳng hạn:

<h1 style="color:blue;">This is a heading</h1>  
<p style="color:red;">This is a paragraph.</p>

Kích thước phông chữ HTML

Thuộc tính kích thước phông chữ xác định kích thước văn bản cho một phần tử HTML:

<h1 style="font-size:300%;">This is a heading</h1>  
<p style="font-size:160%;">This is a paragraph.</p>

Căn chỉnh văn bản HTML

Thuộc tính căn chỉnh văn bản xác định căn chỉnh văn bản ngang cho một thành phần HTML

<h1 style="text-align:center;">Centered Heading</h1>  
<p style="text-align:center;">Centered paragraph.</p>

### Định dạng văn bản HTML

Các phần tử định dạng HTML

Trong phần trước, ta đã tìm hiểu về thuộc tính style HTML. Ngoài ra, HTML cũng định nghĩa các thành phần đặc biệt để xác định văn bản có ý nghĩa nhấn mạnh.

HTML sử dụng các thành phần như <b> và <i> để định dạng văn bản đầu ra là in đậm hoặc in nghiêng.

Các thành phần định dạng được thiết kế để hiển thị các loại văn bản đặc biệt, cụ thể là

<b> - Chữ in đậm

<strong> - Văn bản quan trọng

<i> - Văn bản in nghiêng

<em> - Văn bản nhấn mạnh

<mark> -Văn bản đánh dấu

<small> - Văn bản nhỏ

<del> - Văn bản xóa

<in> - Văn bản chèn

<sub> - Chỉ số dưới

<sup> - Chỉ số trên

Phần tử HTML <b> và <strong>:

Phần tử HTML <b> HTML xác định văn bản in đậm mà không có bất kỳ tầm quan trọng nào.

Trong khi đó, phần tử <strong> HTML xác định văn bản mạnh, có thêm tầm quan trọng "mạnh" về ngữ nghĩa.

<b>This text is bold</b>

<strong>This text is strong</strong>

Phần tử HTML <i> và <em>

Phần tử <i> HTML xác định văn bản in nghiêng mà không có bất kỳ tầm quan trọng nào còn phần tử <em> HTML xác định văn bản được nhấn mạnh, với tầm quan trọng ngữ nghĩa được thêm vào.

Chú ý: Trình duyệt hiển thị <strong> giống <b> và <em> giống <i>. Tuy nhiên, có một sự khác biệt về ý nghĩa của các thẻ này: <b> và <i> định nghĩa văn bản in đậm và in nghiêng, nhưng <strong> và <em> có nghĩa là văn bản là "quan trọng".

Phần tử HTML <small>

Phần tử <small> trong HTML định nghĩa văn bản nhỏ hơn

Phần tử HTML <del>

Phần tử <dell > trong HTML định nghĩa văn bản đã xóa

<p>My favorite color is <del>blue</del> red.</p>

Phần tử HTML <in>

Phần tử <in> HTML xác định văn bản được chèn (thêm).

<p>My favorite <ins>color</ins> is red.</p>

Phần tử HTML <sub>

Phần tử <sub> HTML xác định chỉ số dưới của văn bản.

Phần tử HTML <sup>

Phần tử <sup> HTML xác định chỉ số trên của văn bản.

### HTML Quotations

Các phần tử trích dẫn Quotation và Citation

Giả sử đây là một trích dẫn từ WWF's website: “For 50 years, WWF has been protecting the future of nature. The world's leading conservation organization, WWF works in 100 countries and is supported by 1.2 million members in the United States and close to 5 million globally”

HTML dùng thẻ <q> cho các trích dẫn ngắn. Phần tử <q> HTML xác định một trích dẫn ngắn. Các trình duyệt thường chèn dấu ngoặc kép quanh phần tử <q>. Chẳng hạn:

<p>WWF's goal is to: <q>Build a future where people live in harmony with nature.</q></p>

HTML <blockquote> for Quotations

Phần tử <blockquote> HTML xác định một phần được trích dẫn từ một nguồn khác. Các trình duyệt thường thụt lề các phần tử <blockquote>. Ví dụ:

<p>Here is a quote from WWF's website:</p>  
<blockquote cite="http://www.worldwildlife.org/who/index.html">  
For 50 years, WWF has been protecting the future of nature.  
The world's leading conservation organization,  
WWF works in 100 countries and is supported by  
1.2 million members in the United States and  
close to 5 million globally.  
</blockquote>

HTML <abbr> for Abbreviations

Phần tử <abbr> HTML định nghĩa một từ viết tắt hoặc các chữ viết tắt.

Đánh dấu từ và chữ viết tăt có thể cung cấp thông tin hữu ích cho các trình duyệt, hệ thống dịch thuật và công cụ tìm kiếm.

<p>The <abbr title="World Health Organization">WHO</abbr> was founded in 1948.</p>

HTML <address> for Contact Information

The HTML <address> element defines contact information (author/owner) of a document or an article.

The <address> element is usually displayed in italic. Most browsers will add a line break before and after the element.

Phần tử <address> HTML xác định thông tin liên hệ (tác giả / chủ sở hữu) của tài liệu hoặc bài viết. Phần tử <address> thường được hiển thị bằng chữ nghiêng. Hầu hết các trình duyệt sẽ thêm ngắt dòng trước và sau phần tử.

<address>  
Written by John Doe.<br>   
Visit us at:<br>  
Example.com<br>  
Box 564, Disneyland<br>  
USA  
</address>

HTML <cite> for Work Title

Phần tử <cite> HTML xác định tiêu đề của công việc. Trình duyệt thường hiển thị các phần tử <cite> in nghiêng.

<p><cite>The Scream</cite> by Edvard Munch. Painted in 1893.</p>

The HTML <bdo> element defines bi-directional override.

The HTML <bdo> element defines bi-directional override.

The <bdo> element is used to override the current text direction:

<bdo dir="rtl">This text will be written from right to left</bdo>

### HTML Comments

Thẻ Comment được sử dụng để chèn nhận xét trong mã nguồn HTML. Có thể thêm nhận xét vào nguồn HTML của mình bằng cách sử dụng cú pháp sau:

<!-- Write your comments here -->

Lưu ý:

* Có một dấu chấm than (!) trong thẻ mở, nhưng không có trong thẻ đóng.
* Comment không được trình duyệt hiển thị, nhưng chúng có thể giúp ghi lại mã nguồn HTML của bạn với các bình luận, bạn có thể đặt thông báo và lời nhắc trong HTML của mình:

<!-- This is a comment -->  
<p>This is a paragraph.</p>  
<!-- Remember to add more information here -->

### HTML Colors

Định nghĩa: Màu HTML được chỉ định bằng cách sử dụng tên màu được xác định trước hoặc các giá trị RGB, HEX, HSL, RGBA, HSLA.

* Tên màu:

Trong HTML, một màu có thể được chỉ định bằng cách sử dụng tên màu:

Ví dụ:

<!DOCTYPE html>

<html>

<body>

<h1 style="background-color:Tomato;">Tomato</h1>

<h1 style="background-color:Orange;">Orange</h1>

<h1 style="background-color:DodgerBlue;">DodgerBlue</h1>

<h1 style="background-color:MediumSeaGreen;">MediumSeaGreen</h1>

<h1 style="background-color:Gray;">Gray</h1>

<h1 style="background-color:SlateBlue;">SlateBlue</h1>

<h1 style="background-color:Violet;">Violet</h1>

<h1 style="background-color:LightGray;">LightGray</h1>

</body>

</html>



* Màu nền

Bạn có thể đặt màu nền cho các thành phần HTML:

Ví dụ:

<h1 style="background-color:DodgerBlue;">Hello World</h1>  
<p style="background-color:Tomato;">Lorem ipsum...</p>

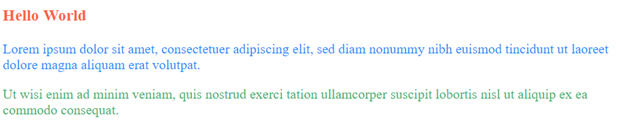
HIỂN THỊ

* Văn bản màu.

Bạn có thể đặt màu của văn bản:

Ví dụ:

<h1 style="color:Tomato;">Hello World</h1>  
<p style="color:DodgerBlue;">Lorem ipsum...</p>  
<p style="color:MediumSeaGreen;">Ut wisi enim...</p>

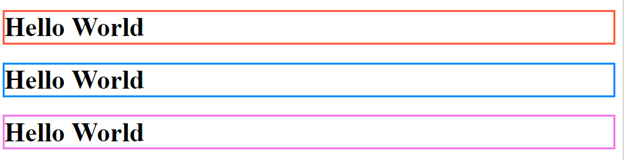


* Màu viền

Bạn có thể đặt màu của đường viền:

Ví dụ:

<h1 style="border:2px solid Tomato;">Hello World</h1>  
<h1 style="border:2px solid DodgerBlue;">Hello World</h1>  
<h1 style="border:2px solid Violet;">Hello World</h1>

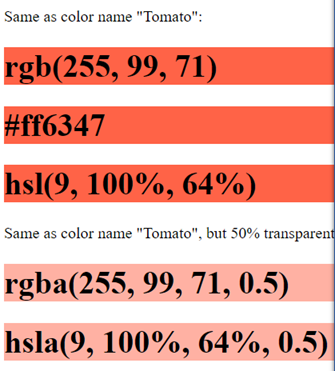


* Giá trị màu

Trong HTML, màu sắc cũng có thể được chỉ định bằng các giá trị RGB, giá trị HEX, giá trị HSL, giá trị RGBA và giá trị HSLA

Ví dụ:

<h1 style="background-color:rgb(255, 99, 71);">...</h1>  
<h1 style="background-color:#ff6347;">...</h1>  
<h1 style="background-color:hsl(9, 100%, 64%);">...</h1>  
<h1 style="background-color:rgba(255, 99, 71, 0.5);">...</h1>  
<h1 style="background-color:hsla(9, 100%, 64%, 0.5);">...</h1>



* Giá trị RGB

Trong HTML, màu có thể được chỉ định làm giá trị RGB, sử dụng công thức này: rgb ( đỏ, xanh lá cây , xanh dương )

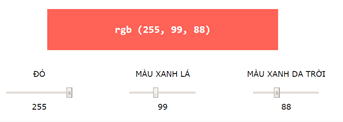
Mỗi tham số (đỏ, xanh lá cây và xanh dương) xác định cường độ của màu trong khoảng từ 0 đến 255.

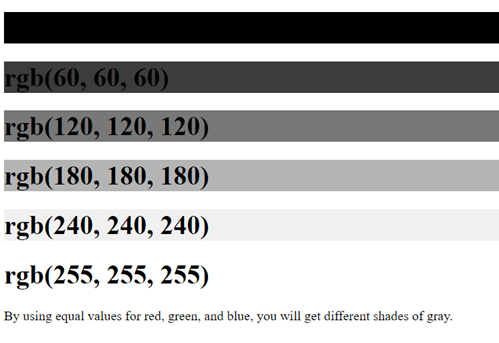
Ví dụ: rgb (255, 0, 0) được hiển thị màu đỏ, vì màu đỏ được đặt thành giá trị cao nhất (255) và các giá trị khác được đặt thành 0.

Để hiển thị màu đen, đặt tất cả các tham số màu thành 0, như thế này: rgb (0, 0, 0).

Để hiển thị màu trắng, đặt tất cả các tham số màu thành 255, như thế này: rgb (255, 255, 255).

Thử nghiệm bằng cách trộn các giá trị RGB bên dưới:





* Giá trị HEX

Trong HTML, một màu có thể được chỉ định bằng cách sử dụng giá trị thập lục phân ở dạng:

**# rrggbb**

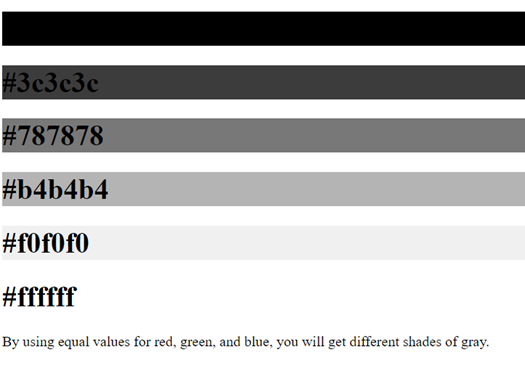
Trong đó rr (đỏ), gg (xanh lá cây) và bb (xanh lam) là các giá trị thập lục phân trong khoảng từ 00 đến ff (giống như số thập phân 0-255).

Ví dụ: # ff0000 được hiển thị màu đỏ, vì màu đỏ được đặt thành giá trị cao nhất (ff) và các giá trị khác được đặt thành giá trị thấp nhất (00).

Ví dụ:



Các sắc thái của màu xám thường được xác định bằng các giá trị bằng nhau cho cả 3 nguồn sáng:



* Gía trị HSL

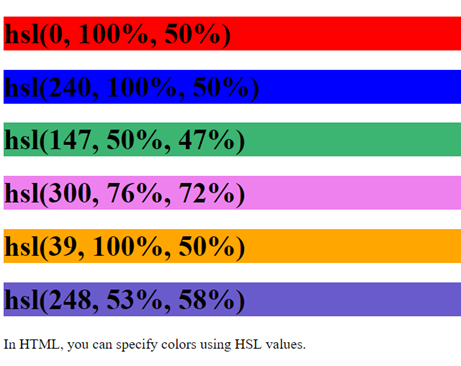
Trong HTML, một màu có thể được chỉ định bằng cách sử dụng màu sắc, độ bão hòa và độ sáng (HSL) ở dạng:

**hsl ( màu sắc , độ bão hòa , độ sáng )**

Hue là một độ trên bánh xe màu từ 0 đến 360. 0 là màu đỏ, 120 là màu xanh lá cây và 240 là màu xanh lam.

Độ bão hòa là một giá trị phần trăm, 0% có nghĩa là màu xám và 100% là màu đầy đủ.

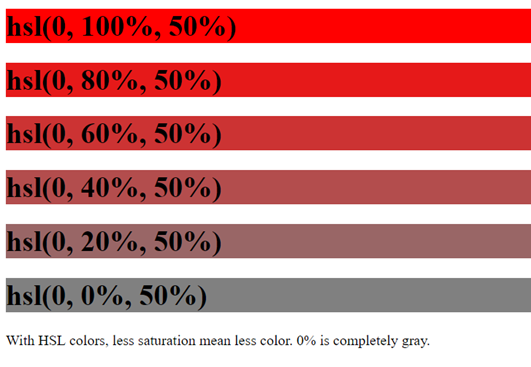
Độ sáng cũng là một tỷ lệ phần trăm, 0% là màu đen, 50% không phải là sáng hay tối, 100% là màu trắng



Độ bão hòa

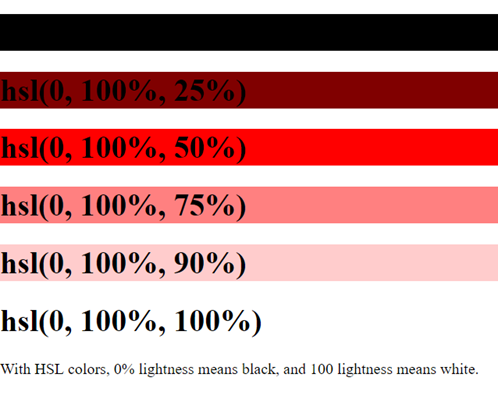
Độ bão hòa có thể được mô tả là cường độ của một màu.100% là màu thuần khiết, không có sắc độ xám

50% là 50% màu xám, nhưng bạn vẫn có thể nhìn thấy màu.0% là màu xám hoàn toàn, bạn không thể nhìn thấy màu nữa.

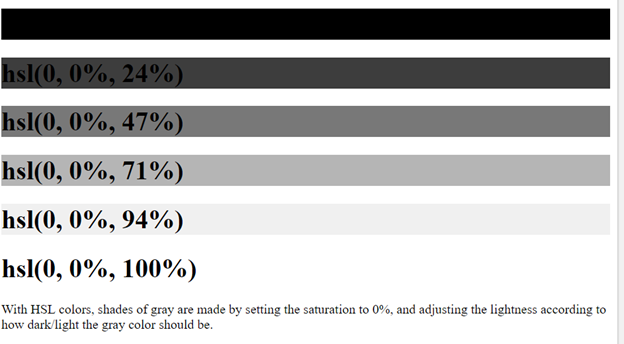


Nhẹ nhàng

Độ sáng của màu sắc có thể được mô tả là bạn muốn cung cấp bao nhiêu ánh sáng, trong đó 0% có nghĩa là không có ánh sáng (đen), 50% có nghĩa là 50% ánh sáng (không tối cũng không sáng) 100% có nghĩa là ánh sáng đầy đủ (trắng).



Màu xám thường được xác định bằng cách đặt màu sắc và độ bão hòa thành 0 và điều chỉnh độ sáng từ 0% đến 100% để có được màu tối hơn / sáng hơn:

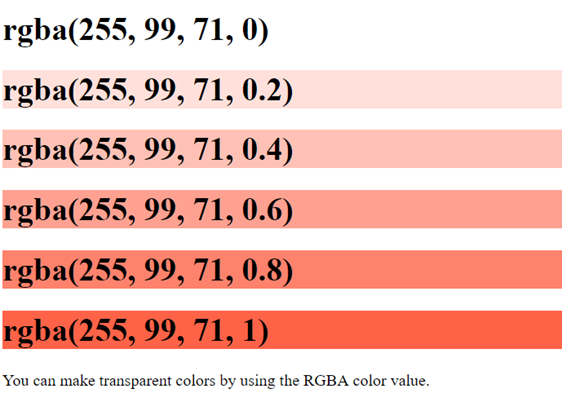


* Giá trị RGBA

Giá trị màu RGBA là phần mở rộng của giá trị màu RGB với kênh alpha - chỉ định độ mờ cho màu.Giá trị màu RGBA được chỉ định bằng:

**rgba ( đỏ, xanh lá cây , xanh dương, alpha )**

Tham số alpha là một số trong khoảng 0,0 (hoàn toàn trong suốt) và 1.0 (hoàn toàn không minh bạch)

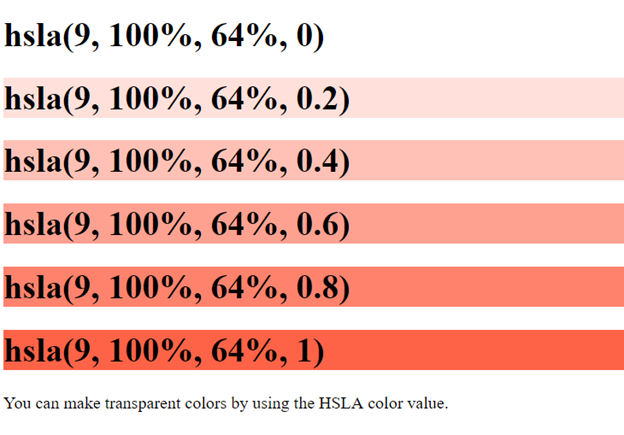


* Giá trị HSLA

Giá trị màu HSLA là phần mở rộng của giá trị màu HSL với kênh alpha - chỉ định độ mờ cho màu. Giá trị màu HSLA được chỉ định bằng:

**hsla ( màu sắc, độ bão hòa , độ sáng, alpha )**

Tham số alpha là một số trong khoảng 0,0 (hoàn toàn trong suốt) và 1.0 (hoàn toàn không minh bạch):



### HTML CSS

* Tạo kiểu HTML bằng CSS

CSS là viết tắt của C ascading S tyle S heets.

CSS mô tả cách các phần tử HTML được hiển thị trên màn hình, giấy hoặc trong các phương tiện khác .

CSS tiết kiệm rất nhiều công việc . Nó có thể kiểm soát bố cục của nhiều trang web cùng một lúc.

CSS có thể được thêm vào các phần tử HTML theo 3 cách:

Inline - bằng cách sử dụng thuộc tính style trong các phần tử HTML

Nội bộ - bằng cách sử dụng một <style>yếu tố trong <head>phần

Bên ngoài - bằng cách sử dụng tệp CSS bên ngoài

Cách phổ biến nhất để thêm CSS, là giữ các kiểu trong các tệp CSS riêng biệt. Tuy nhiên, ở đây chúng tôi sẽ sử dụng kiểu dáng nội tuyến và nội bộ, bởi vì điều này dễ thể hiện hơn và dễ dàng hơn để bạn tự thử.

* CSS nội tuyến

Ví dụ:

<h1 style="color:blue;">This is a Blue Heading</h1>



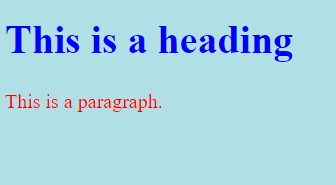
* CSS nội bộ

Một CSS nội bộ được sử dụng để xác định kiểu cho một trang HTML.

CSS nội bộ được xác định trong <head>phần của trang HTML, bên trong một <style>phần tử:

Ví dụ:

<!DOCTYPE html>  
<html>  
<head>  
<style>  
body {background-color: powderblue;}  
h1   {color: blue;}  
p    {color: red;}  
</style>  
</head>  
<body>  
  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>  
  
</body>  
</html>



* CSS bên ngoài

Một biểu định kiểu ngoài được sử dụng để xác định kiểu cho nhiều trang HTML.

Với biểu định kiểu bên ngoài, bạn có thể thay đổi giao diện của toàn bộ trang web bằng cách thay đổi một tệp!

Để sử dụng biểu định kiểu bên ngoài, hãy thêm một liên kết đến nó trong <head>phần của trang HTML:

Ví dụ:

<!DOCTYPE html>  
<html>  
<head>  
  <link rel="stylesheet" href="styles.css">  
</head>  
<body>  
  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>  
  
</body>  
</html>

Một biểu định kiểu bên ngoài có thể được viết trong bất kỳ trình soạn thảo văn bản nào. Tệp không được chứa bất kỳ mã HTML nào và phải được lưu với phần mở rộng .css.

body {  
  background-color: powderblue;  
}  
h1 {  
  color: blue;  
}  
p {  
  color: red;  
}

* Phông chữ CSS

Thuộc tính CSS Color xác định màu văn bản sẽ được sử dụng.

Thuộc tính CSS font-family xác định phông chữ sẽ được sử dụng.

Thuộc tính CSS font-size  xác định kích thước văn bản sẽ được sử dụng.

<!DOCTYPE html>  
<html>  
<head>  
<style>  
h1 {  
  color: blue;  
  font-family: verdana;  
  font-size: 300%;  
}  
p  {  
  color: red;  
  font-family: courier;  
  font-size: 160%;  
}  
</style>  
</head>  
<body>  
  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>  
  
</body>  
</html>

* Biên giới CSS

Thuộc tính CSS border xác định đường viền xung quanh thành phần HTML:

Ví dụ:

p {  
  border: 1px solid powderblue;  
}

* Đệm CSS

Thuộc tính Padding CSS xác định phần đệm (khoảng trắng) giữa văn bản và đường viền

Ví dụ:

p {  
  border: 1px solid powderblue;  
  padding: 30px;  
}

* Ký quỹ CSS

Thuộc tính CSS **margin** xác định lề (dấu cách) bên ngoài đường viền:

p {  
  border: 1px solid powderblue;  
  margin: 50px;  
}

* Thuộc tính id

Để xác định một kiểu cụ thể cho một phần tử đặc biệt, hãy thêm một idthuộc tính cho phần tử:

<p id="p01">I am different</p>

sau đó xác định kiểu cho phần tử với id cụ thể:

#p01 {  
  color: blue;  
}

* Thuộc tính lớp

Để xác định kiểu cho các loại phần tử đặc biệt, hãy thêm classthuộc tính cho phần tử:

<p class="error">I am different</p>

sau đó định nghĩa một kiểu cho các phần tử với lớp cụ thể:

Ví dụ:

p.error {  
  color: red;  
}

* **Tóm tắt chương**

Sử dụng stylethuộc tính HTML để tạo kiểu nội tuyến

Sử dụng stylephần tử HTML để xác định CSS nội bộ

Sử dụng <link> phần tử HTML để chỉ tệp CSS bên ngoài

Sử dụng <head>phần tử HTML để lưu trữ các phần tử stylevà <link>

Sử dụng thuộc tính CSS colorcho màu văn bản

Sử dụng thuộc tính CSS font-familycho phông chữ văn bản

Sử dụng thuộc tính CSS font-sizecho kích thước văn bản

Sử dụng thuộc tính CSS bordercho viền

Sử dụng thuộc tính CSS paddingcho không gian bên trong đường viền

Sử dụng thuộc tính CSS margincho không gian bên ngoài đường viền

### HTML Links

Liên kết được tìm thấy trong gần như tất cả các trang web. Liên kết cho phép người dùng nhấp theo cách của họ từ trang này sang trang khác.

* Liên kết HTML là siêu liên kết.

Bạn có thể nhấp vào một liên kết và chuyển đến một tài liệu khác.

Khi bạn di chuyển chuột qua một liên kết, mũi tên chuột sẽ biến thành một bàn tay nhỏ.

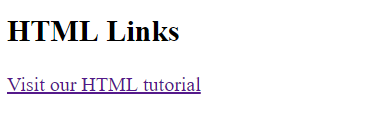
Lưu ý : một liên kết không phải là một văn bản. nó có thể là bất kỳ một yếu tố nào khác

* Liên kết HTML - Cú pháp

Các siêu liên kết được xác định bằng <a>thẻ HTML :

<a href="url">link text</a>

<a href="https://www.w3schools.com/html/">Visit our HTML tutorial</a>



Các hrefthuộc tính xác định địa chỉ đích ( https://www.w3schools.com/html/ ) của liên kết.

Các văn bản liên kết là phần nhìn thấy được (Thăm hướng dẫn HTML của chúng tôi).

Nhấp vào văn bản liên kết sẽ đưa bạn đến địa chỉ được chỉ định.

Lưu ý: Không có dấu gạch chéo ở cuối địa chỉ thư mục con, bạn có thể tạo hai yêu cầu đến máy chủ. Nhiều máy chủ sẽ tự động thêm dấu gạch chéo vào cuối địa chỉ và sau đó tạo một yêu cầu mới.

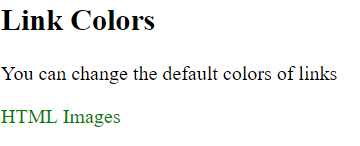
* Liên kết địa phương.

Ví dụ trên đã sử dụng một URL tuyệt đối (một địa chỉ web đầy đủ).

Liên kết cục bộ (liên kết đến cùng một trang web) được chỉ định bằng một URL tương đối (không có https: // www ....).

Ví dụ:

<a href="html\_images.asp">HTML Images</a>:



* Màu sắc liên kết HTML

Theo mặc định, một liên kết sẽ xuất hiện như thế này (trong tất cả các trình duyệt):

* Một liên kết không mong muốn được gạch chân và màu xanh
* Một liên kết truy cập được gạch chân và màu tím
* Một liên kết hoạt động được gạch chân và màu đỏ

Bạn có thể thay đổi màu mặc định bằng cách sử dụng CSS:

Ví dụ:

<style>  
a:link {  
  color: green;  
  background-color: transparent;  
  text-decoration: none;  
}  
  
a:visited {  
  color: pink;  
  background-color: transparent;  
  text-decoration: none;  
}  
  
a:hover {  
  color: red;  
  background-color: transparent;  
  text-decoration: underline;  
}  
  
a:active {  
  color: yellow;  
  background-color: transparent;  
  text-decoration: underline;  
}  
</style>

Liên kết thường được tạo kiểu như các nút, bằng cách sử dụng CSS:

Ví dụ:

<style>  
a:link, a:visited {  
  background-color: #f44336;  
  color: white;  
  padding: 15px 25px;  
  text-align: center;  
  text-decoration: none;  
  display: inline-block;  
}  
  
a:hover, a:active {  
  background-color: red;  
}  
</style>

* Liên kết HTML - Thuộc tính mục tiêu

Các target thuộc tính xác định nơi để mở tài liệu được liên kết.

Các target thuộc tính có thể có một trong các giá trị sau:

- Black – mở tài liệu được liên kết trong cửa sổ hoặc tab mới

- Self \_ mở tài liệu được liên kết trong cùng một cửa sổ/tab khi được nhấp(đây là mặc định)

- Parent – mở tài liệu được liên kết trong toàn bộ cửa sổ

- Framename\_ mở tài liệu được liên kết trong một khung có tên

Ví dụ:

<a href="https://www.w3schools.com/" target="\_blank">Visit W3Schools!</a>

Mẹo: Nếu trang web của bạn bị khóa trong một khung, bạn có thể sử dụng target="\_top"để thoát ra khỏi khung:

<a href="https://www.w3schools.com/html/" target="\_top">HTML5 tutorial!</a>

* Lên kết HTML - Hình ảnh dưới dạng Liên kết

Ví dụ:

<a href="default.asp">  
  <img src="smiley.gif" alt="HTML tutorial" style="width:42px;height:42px;border:0;">  
</a>

Lưu ý:border:0; được thêm vào để ngăn IE9 (và trước đó) hiển thị đường viền xung quanh hình ảnh (khi hình ảnh là một liên kết).

* Tiêu đề liên kết

Các title thuộc tính xác định thông tin thêm về một phần tử. Thông tin thường được hiển thị dưới dạng văn bản chú giải công cụ khi chuột di chuyển qua phần tử.

Ví dụ:

Các titlethuộc tính xác định thông tin thêm về một phần tử. Thông tin thường được hiển thị dưới dạng văn bản chú giải công cụ khi chuột di chuyển qua phần tử.

<a href="https://www.w3schools.com/html/" title="Go to W3Schools HTML section">Visit our HTML Tutorial</a>

* Liên kết HTML - Tạo một Bookmark

Dấu trang HTML được sử dụng để cho phép người đọc chuyển đến các phần cụ thể của trang Web.

Dấu trang có thể hữu ích nếu trang web của bạn rất dài.

Để tạo dấu trang, trước tiên bạn phải tạo dấu trang và sau đó thêm một liên kết đến nó.

Khi liên kết được nhấp, trang sẽ cuộn đến vị trí có dấu trang.

Ví dụ:

Đầu tiên, tạo một dấu trang với idthuộc tính:

<h2 id="C4">Chapter 4</h2>

Sau đó, thêm một liên kết đến dấu trang ("Chuyển đến Chương 4"), từ trong cùng một trang:

<a href="#C4">Jump to Chapter 4</a>

Hoặc, thêm một liên kết đến dấu trang ("Chuyển đến Chương 4"), từ một trang khác:

<a href="html\_demo.html#C4">Jump to Chapter 4</a>

* Đường dẫn bên ngoài

Các trang bên ngoài có thể được tham chiếu với một URL đầy đủ hoặc với một đường dẫn liên quan đến trang web hiện tại.

Ví dụ này sử dụng URL đầy đủ để liên kết đến một trang web:

<a href="https://www.w3schools.com/html/default.asp">HTML tutorial</a>

Ví dụ này liên kết đến một trang nằm trong thư mục html trên trang web hiện tại:

<a href="/html/default.asp">HTML tutorial</a>

* Tóm tắt chương

- Sử dụng <a>phần tử để xác định một liên kết

- Sử dụng hrefthuộc tính để xác định địa chỉ liên kết

- Sử dụng targetthuộc tính để xác định nơi mở tài liệu được liên kết

- Sử dụng <img>yếu tố (bên trong <a>) để sử dụng hình ảnh làm liên kết

- Sử dụng idthuộc tính (id = " value ") để xác định dấu trang trong trang

- Sử dụng hrefthuộc tính (href = "# value ") để liên kết đến dấu trang

### HTML5

#### Giới thiệu

#### Có gì mới trong HTML5?

Khai báo DOCTYPE cho HTML5 rất đơn giản:

<!DOCTYPE html>

Khai báo mã hóa ký tự (bộ ký tự) cũng rất đơn giản:

<meta charset="UTF-8">

Ví dụ HTML5:

<!DOCTYPE html>  
<html>  
<head>  
<meta charset="UTF-8">  
<title>Title of the document</title>  
</head>  
<body>  
Content of the document......  
</body>  
</html>

Mã hóa ký tự mặc định trong HTML5 là UTF-8.

#### Các yếu tố mới trong html mới

Các yếu tố HTML5 mới thú vị nhất là:

New yếu tố ngữ nghĩa như <header>, <footer>, <article>, <section>

Các thuộc tính mới của các thành phần biểu mẫu như số, ngày, thời gian, lịch và phạm vi.

Yếu tố đồ họa mới : <svg>, <canvas>.

Các yếu tố đa phương tiện mới : <audio>, <video>.

Trong chương tiếp theo, Hỗ trợ HTML5 , bạn sẽ tìm hiểu cách "dạy" các trình duyệt cũ hơn để xử lý các phần tử HTML "không xác định" (mới).

#### API HTML5 mới (Giao diện lập trình ứng dụng)

Các API mới thú vị nhất trong HTML5 là:

- Định vị địa lý HTML

- Kéo và thả HTML

- Lưu trữ cục bộ HTML

- Bộ nhớ cache ứng dụng HTML

- Công nhân web HTML

- HTML SSE

#### Các phần tử bị xóa trong HTML5

Các phần tử HTML4 sau đây đã bị xóa trong HTML5:

Trong chương Di chuyển HTML5 , bạn sẽ tìm hiểu cách dễ dàng di chuyển từ HTML4 sang HTML5.

#### Lịch sử HTML

Kể từ những ngày đầu của World Wide Web, đã có nhiều phiên bản HTML.

#### HTML5 hỗ trợ

#### Hỗ trợ trình duyệt HTML5

HTML5 được hỗ trợ trong tất cả các trình duyệt hiện đại.

Ngoài ra, tất cả các trình duyệt, cũ và mới, tự động xử lý các phần tử không được nhận dạng dưới dạng phần tử nội tuyến.

Do đó, bạn có thể "dạy" các trình duyệt cũ hơn để xử lý các phần tử HTML "không xác định".

#### Xác định các phần tử ngữ nghĩa là các phần tử khối

HTML5 định nghĩa tám yếu tố ngữ nghĩa mới . Tất cả đều là các yếu tố cấp khối .

Để bảo đảm hành vi chính xác trong các trình duyệt cũ hơn, bạn có thể đặt thuộc tính hiển thị CSS cho các thành phần HTML này thành chặn :

header, section, footer, aside, nav, main, article, figure {  
  display: block;  
}

#### Thêm các yếu tố mới vào HTML

Bạn cũng có thể thêm các yếu tố mới vào trang HTML bằng thủ thuật trình duyệt.

Ví dụ này thêm một yếu tố mới được gọi <myHero>vào trang HTML và xác định kiểu cho nó:

<!DOCTYPE html>  
<html>  
<head>  
<script>document.createElement("myHero")</script>  
<style>  
myHero {  
  display: block;  
  background-color: #dddddd;  
  padding: 50px;  
  font-size: 30px;  
}  
</style>  
</head>  
<body>  
  
<h1>A Heading</h1>  
<myHero>My Hero Element</myHero>  
  
</body>  
</html>

Câu lệnh JavaScript document.createElement("myHero")là cần thiết để tạo một phần tử mới trong IE 9 và trước đó.

#### Sự cố với Internet Explorer 8

#### Cú pháp cho HTML5Shiv

HTML5Shiv được đặt trong <head>thẻ.

HTML5Shiv là một tệp javascript được tham chiếu trong <script>thẻ.

Bạn nên sử dụng HTML5Shiv khi bạn đang sử dụng các yếu tố HTML5 mới như: <article>, <section>, <aside>, <nav>, <footer>.

Bạn có thể tải xuống phiên bản HTML5shiv mới nhất từ ​​github hoặc tham khảo phiên bản CDN tại https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js

Cú pháp

<head>  
  <!--[if lt IE 9]>  
    <script src="/js/html5shiv.js"></script>  
  <![endif]-->  
</head>

Ví dụ HTML5Shiv

Nếu bạn không muốn tải xuống và lưu trữ HTML5Shiv trên trang web của mình, bạn có thể tham khảo phiên bản được tìm thấy trên trang web CDN.

Tập lệnh HTML5Shiv phải được đặt trong <head>phần tử, sau bất kỳ biểu định kiểu nào:

Ví dụ:

<!DOCTYPE html>  
<html>  
<head>  
<meta charset="UTF-8">  
<!--[if lt IE 9]>  
  <script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>  
<![endif]-->  
</head>  
<body>  
<section>  
<h1>Famous Cities</h1>  
<article>  
<h2>London</h2>  
<p>London is the capital city of England. It is the most populous city in the United Kingdom, with a metropolitan area of over 13 million inhabitants.</p>  
</article>  
<article>  
<h2>Paris</h2>  
<p>Paris is the capital and most populous city of France.</p>  
</article>  
<article>  
<h2>Tokyo</h2>  
<p>Tokyo is the capital of Japan, the center of the Greater Tokyo Area, and the most populous metropolitan area in the world.</p>  
</article>  
</section>  
  
</body>  
</html>

#### Các yếu tố mới trong HTML

#### Các yếu tố mới trong HTML5

#### Các yếu tố cấu trúc / ngữ nghĩa mới

HTML5 cung cấp các yếu tố mới cho cấu trúc tài liệu tốt hơn:

|  |  |
| --- | --- |
| Tag | Description |
| <article> | Defines an article in a document |
| <aside> | Defines content aside from the page content |
| [<bdi>](https://www.w3schools.com/tags/tag_bdi.asp) | Isolates a part of text that might be formatted in a different direction from other text outside it |
| [<details>](https://www.w3schools.com/tags/tag_details.asp) | Defines additional details that the user can view or hide |
| [<dialog>](https://www.w3schools.com/tags/tag_dialog.asp) | Defines a dialog box or window |
| [<figcaption>](https://www.w3schools.com/tags/tag_figcaption.asp) | Defines a caption for a <figure> element |
| [<figure>](https://www.w3schools.com/tags/tag_figure.asp) | Defines self-contained content |
| [<footer>](https://www.w3schools.com/tags/tag_footer.asp) | Defines a footer for a document or section |
| [<header>](https://www.w3schools.com/tags/tag_header.asp) | Defines a header for a document or section |
| [<main>](https://www.w3schools.com/tags/tag_main.asp) | Defines the main content of a document |
| [<mark>](https://www.w3schools.com/tags/tag_mark.asp) | Defines marked/highlighted text |
| [<meter>](https://www.w3schools.com/tags/tag_meter.asp) | Defines a scalar measurement within a known range (a gauge) |
| [<nav>](https://www.w3schools.com/tags/tag_nav.asp) | Defines navigation links |
| [<progress>](https://www.w3schools.com/tags/tag_progress.asp) | Represents the progress of a task |
| [<rp>](https://www.w3schools.com/tags/tag_rp.asp) | Defines what to show in browsers that do not support ruby annotations |
| [<rt>](https://www.w3schools.com/tags/tag_rt.asp) | Defines an explanation/pronunciation of characters (for East Asian typography) |
| [<ruby>](https://www.w3schools.com/tags/tag_ruby.asp) | Defines a ruby annotation (for East Asian typography) |
| [<section>](https://www.w3schools.com/tags/tag_section.asp) | Defines a section in a document |
| [<summary>](https://www.w3schools.com/tags/tag_summary.asp) | Defines a visible heading for a <details> element |
| [<time>](https://www.w3schools.com/tags/tag_time.asp) | Defines a date/time |
| [<wbr>](https://www.w3schools.com/tags/tag_wbr.asp) | Defines a possible line-break |

#### Yếu tố hình thức mới

|  |  |
| --- | --- |
| Tag | Description |
| <datalist> | Specifies a list of pre-defined options for input controls |
| <output> | Defines the result of a calculation |

#### Kiểu đầu vào mới.

|  |  |
| --- | --- |
| Kiểu đầu vào mới | Thuộc tính đầu vào mới |
| màu sắc  ngày  ngày giờ  datetime-local  e-mail  tháng  con số  phạm vi  Tìm kiếm  điện thoại  thời gian  url  tuần | tự động hoàn thành  tự động lấy nét  hình thức  hình thành  formenctype  hình thức  formnovalidate  nhắm mục tiêu  chiều cao và chiều rộng  danh sách  tối thiểu và tối đa  nhiều  mô hình (regrec)  giữ chỗ  cần thiết  bươc |

#### HTML5 cú pháp và thuộc tính mới.

HTML5 cho phép bốn cú pháp khác nhau cho các thuộc tính.

Ví dụ này cho thấy các cú pháp khác nhau được sử dụng trong một <input>thẻ:

|  |  |
| --- | --- |
| Type | Example |
| Empty | <input type="text" value="John" **disabled**> |
| Unquoted | <input type="text" **value=John**> |
| Double-quoted | <input type="text" **value="John Doe"**> |
| Single-quoted | <input type="text" **value='John Doe'**> |

Trong HTML5, tất cả bốn cú pháp có thể được sử dụng, tùy thuộc vào những gì cần thiết cho thuộc tính.

#### Đồ họa HTML5

|  |  |
| --- | --- |
| Tag | Description |
| <canvas> | Draw graphics, on the fly, via scripting (usually JavaScript) |
| <svg> | Draw scalable vector graphics |

#### Yếu tố truyền thông mới

|  |  |
| --- | --- |
| Tag | Description |
| <audio> | Defines sound content |
| <embed> | Defines a container for an external (non-HTML) application |
| <source> | Defines multiple media resources for media elements (<video> and <audio>) |
| <track> | Defines text tracks for media elements (<video> and <audio>) |
| <video> | Defines video or movie |

### HTML Images

Cú pháp hình ảnh HTML

Trong HTML, hình ảnh được xác định bằng <img>

Các <img>thẻ trống, nó chỉ chứa các thuộc tính, và không có một thẻ đóng.

Các srcthuộc tính xác định URL (địa chỉ web) của hình ảnh:

<img src="url">

Thuộc tính alt

Các altthuộc tính cung cấp một văn bản thay thế cho hình ảnh, nếu người sử dụng đối với một số lý do không thể xem nó (vì kết nối chậm, một lỗi trong các thuộc tính src, hoặc nếu người dùng sử dụng một trình đọc màn hình).

Giá trị của altthuộc tính sẽ mô tả hình ảnh:

Thí dụ

**<img** src="img\_chania.jpg" alt="Flowers in Chania"**>**

Nếu trình duyệt không thể tìm thấy hình ảnh, nó sẽ hiển thị giá trị của alt thuộc tính:

Lưu ý: Các altthuộc tính là bắt buộc. Một trang web sẽ không xác nhận chính xác mà không có nó.

Kích thước hình ảnh - Chiều rộng và chiều cao

Bạn có thể sử dụng stylethuộc tính để chỉ định chiều rộng và chiều cao của hình ảnh.

Thí dụ

**<img** src="img\_girl.jpg" alt="Girl in a jacket" style="width:500px;height:600px;"**>**

Ngoài ra, bạn có thể sử dụng các thuộc tính width và height:

Thí dụ

**<img** src="img\_girl.jpg" alt="Girl in a jacket" width="500" height="600"**>**

Các thuộc tính widthvà heightluôn xác định chiều rộng và chiều cao của hình ảnh bằng pixel.

Lưu ý: Luôn chỉ định chiều rộng và chiều cao của hình ảnh. Nếu chiều rộng và chiều cao không được chỉ định, trang có thể nhấp nháy trong khi hình ảnh tải.

Hình ảnh trong thư mục khác

Nếu không được chỉ định, trình duyệt sẽ tìm thấy hình ảnh trong cùng thư mục với trang web.

Tuy nhiên, thông thường để lưu trữ hình ảnh trong một thư mục phụ. Sau đó, bạn phải bao gồm tên thư mục trong srcthuộc tính:

Thí dụ

**<img** src="/images/html5.gif" alt="HTML5 Icon" style="width:128px;height:128px;"**>**

Hình ảnh trên máy chủ khác

Một số trang web lưu trữ hình ảnh của họ trên các máy chủ hình ảnh.

Trên thực tế, bạn có thể truy cập hình ảnh từ bất kỳ địa chỉ web nào trên thế giới:

Thí dụ

**<img** src="https://www.w3schools.com/images/w3schools\_green.jpg" alt="W3Schools.com"**>**

HTML cho phép GIF động:

Thí dụ

**<img** src="programming.gif" alt="Computer Man" style="width:48px;height:48px;"**>**

Hình ảnh dưới dạng một liên kết

Để sử dụng hình ảnh làm liên kết, hãy đặt <img>thẻ bên trong <a> thẻ:

Thí dụ

**<a** href="default.asp"**>**

**<img** src="smiley.gif" alt="HTML tutorial" style="width:42px;height:42px;border:0;"**>**

**</a>**

Lưu ý: border:0; được thêm vào để ngăn IE9 (và trước đó) hiển thị đường viền xung quanh hình ảnh (khi hình ảnh là một liên kết).

**Hình ảnh nổi**

Sử dụng thuộc tính CSS floatđể cho phép hình ảnh nổi sang phải hoặc bên trái của văn bản:

Thí dụ

**<p><img** src="smiley.gif" alt="Smiley face" style="float:right;width:42px;height:42px;"**>**

The image will float to the right of the text.**</p>**

**<p><img** src="smiley.gif" alt="Smiley face" style="float:left;width:42px;height:42px;"**>**

The image will float to the left of the text.**</p>**

Trình đọc màn hình HTML

Trình đọc màn hình là một chương trình phần mềm đọc mã HTML, chuyển đổi văn bản và cho phép người dùng "nghe" nội dung. Trình đọc màn hình rất hữu ích cho những người khiếm thị hoặc khuyết tật học tập.

### HTML Tables

Các bảng HTML cho phép lập trình viên sắp xếp các dữ liệu như văn bản, hình ảnh, đường link… vào các ô trong bảng.

Bảng HTML được tạo ra bằng cách sử dụng thẻ <table> trong đó: thẻ <tr> được sử dụng để tạo các hàng và thẻ <td> được sử dụng để tạo các ô.

Ví dụ bảng trong HTML

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du bang trong HTML**</title>**

**</head>**

**<body>**

**<table** border="1"**>**

**<tr>**

**<td>**Row 1, Column 1**</td>**

**<td>**Row 1, Column 2**</td>**

**</tr>**

**<tr>**

**<td>**Row 2, Column 1**</td>**

**<td>**Row 2, Column 2**</td>**

**</tr>**

**</table>**

**</body>**

**</html>**

Chạy đoạn mã trên sẽ tạo bảng sau:

|  |  |
| --- | --- |
| Row 1, Column 1 | Row 1, Column 2 |
| Row 2, Column 1 | Row 2, Column 2 |

Tại đây border là một thuộc tính của thẻ <table> và được sử dụng để đặt Border (đường viền) dọc tất cả các ô. Nếu bạn không cần Border, bạn có thể sử dụng border="0".

**Tiêu đề bảng trong HTML**

Tiêu đề bảng có thể được xác định bằng thẻ <th>. Thẻ này để thế chỗ cho thẻ <td> , mà được sử dụng để đại diện cho các ô dữ liệu. Thông thường bạn sẽ đặt hàng đầu tiên của bảng là tiêu đề như hình dưới, ngoài ra bạn có thể sử dụng phần tử <th> trong bất kỳ hàng nào.

Ví dụ

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du tieu de bang**</title>**

**</head>**

**<body>**

**<table** border="1"**>**

**<tr>**

**<th>**Ten nhan vien**</th>**

**<th>**Luong**</th>**

**</tr>**

**<tr>**

**<td>**Minh Chinh**</td>**

**<td>**5000**</td>**

**</tr>**

**<tr>**

**<td>**Duy Manh**</td>**

**<td>**7000**</td>**

**</tr>**

**</table>**

**</body>**

**</html>**

Kết quả hiển thị là:

|  |  |
| --- | --- |
| **Ten nhan vien** | **Luong** |
| Minh Chinh | 5000 |
| Duy Manh | 7000 |

Hai thuộc tính cellpadding và cellspacing trong HTML:

Hai thuộc tính này bạn sử dụng để điều chỉnh khoảng trống trắng trong các ô bảng. Thuộc tính cellspacing xác định độ rộng của Border, trong khi thuộc tính cellpadding thay cho khoảng cách giữa Border ô và nội dung trong một ô.

Thuộc tính colspan và rowspan trong HTML:

Bạn sử dụng thuộc tính colspan để nhập hai hay nhiều cột vào một cột. Tương tự là thuộc tính rowspan để nhập hai hay nhiều hàng vào một hàng.

Ví dụ

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du the colspan va rowspan trong HTML**</title>**

**</head>**

**<body>**

**<table** border="1"**>**

**<tr>**

**<th>**Column 1**</th>**

**<th>**Column 2**</th>**

**<th>**Column 3**</th>**

**</tr>**

**<tr><td** rowspan="2"**>**Row 1 Cell 1**</td><td>**Row 1 Cell 2**</td><td>**Row 1 Cell 3**</td></tr>**

**<tr><td>**Row 2 Cell 2**</td><td>**Row 2 Cell 3**</td></tr>**

**<tr><td** colspan="3"**>**Row 3 Cell 1**</td></tr>**

**</table>**

**</body>**

**</html>**

Kết quả hiển thị là:

|  |  |  |
| --- | --- | --- |
| **Column 1** | **Column 2** | **Column 3** |
| Row 1 Cell 1 | Row 1 Cell 2 | Row 1 Cell 3 |
| Row 2 Cell 2 | Row 2 Cell 3 |
| Row 3 Cell 1 | | |

Xác định background cho bảng trong HTML

Bạn có thể sử dụng một trong hai cách sau để thiết lập nền của bảng:

* Thuộc tính bgcolor - Bạn sử dụng để thiết lập màu nền cho cả bảng hoặc chỉ cho một ô.
* Thuộc tính background - Bạn sử dụng để thiết lập ảnh nền cho cả bảng hoặc chỉ cho một ô.

Bên cạnh đó, bạn có thể thiết lập màu cho Border bằng cách sử dụng thuộc tính bordercolor.

Ví dụ

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du tao background cho bang trong HTML**</title>**

**</head>**

**<body>**

**<table** border="1" bordercolor="green" bgcolor="yellow"**>**

**<tr>**

**<th>**Column 1**</th>**

**<th>**Column 2**</th>**

**<th>**Column 3**</th>**

**</tr>**

**<tr><td** rowspan="2"**>**Row 1 Cell 1**</td><td>**Row 1 Cell 2**</td><td>**Row 1 Cell 3**</td></tr>**

**<tr><td>**Row 2 Cell 2**</td><td>**Row 2 Cell 3**</td></tr>**

**<tr><td** colspan="3"**>**Row 3 Cell 1**</td></tr>**

**</table>**

**</body>**

**</html>**

Nó sẽ tạo ra kết quả sau:

|  |  |  |
| --- | --- | --- |
| **Column 1** | **Column 2** | **Column 3** |
| Row 1 Cell 1 | Row 1 Cell 2 | Row 1 Cell 3 |
| Row 2 Cell 2 | Row 2 Cell 3 |
| Row 3 Cell 1 | | |

Dưới đây là một ví dụ về sử dụng thuộc tính **background**.

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du tao background cho bang trong HTML**</title>**

**</head>**

**<body>**

**<table** border="1" bordercolor="green" background="/images/test.png"**>**

**<tr>**

**<th>**Column 1**</th>**

**<th>**Column 2**</th>**

**<th>**Column 3**</th>**

**</tr>**

**<tr><td** rowspan="2"**>**Row 1 Cell 1**</td><td>**Row 1 Cell 2**</td><td>**Row 1 Cell 3**</td></tr>**

**<tr><td>**Row 2 Cell 2**</td><td>**Row 2 Cell 3**</td></tr>**

**<tr><td** colspan="3"**>**Row 3 Cell 1**</td></tr>**

**</table>**

**</body>**

**</html>**

Kết quả hiển thị là. Ở đây ảnh nền không áp dụng cho Tiêu đề bảng

|  |  |  |
| --- | --- | --- |
| Column 1 | Column 2 | Column 3 |
| Row 1 Cell 1 | Row 1 Cell 2 | Row 1 Cell 3 |
| Row 2 Cell 2 | Row 2 Cell 3 |
| Row 3 Cell 1 | | |

Chiều cao và độ rộng bảng trong HTML

Bạn sử dụng thuộc tính width và height để thiết lập độ rộng và chiều cao của bảng. Bạn có thể thiết lập nó bằng Pixel hoặc phần trăm của khu vực đặt hình.

Ví dụ

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Chieu cao va do rong bang trong HTML**</title>**

**</head>**

**<body>**

**<table** border="1" width="400" height="150"**>**

**<tr>**

**<td>**Row 1, Column 1**</td>**

**<td>**Row 1, Column 2**</td>**

**</tr>**

**<tr>**

**<td>**Row 2, Column 1**</td>**

**<td>**Row 2, Column 2**</td>**

**</tr>**

**</table>**

**</body>**

**</html>**

Nó sẽ tạo ra kết quả sau:

|  |  |
| --- | --- |
| Row 1, Column 1 | Row 1, Column 2 |
| Row 2, Column 1 | Row 2, Column 2 |

Phụ đề bảng trong HTML:

Thẻ caption được sử dụng để lưu giữ một tiêu đề hoặc lời giải thích cho bảng và nó được hiển thị trên đầu của bảng. Thẻ này là thẻ cũ trong các phiên bản mới của HTML/XHTML.

Ví dụ

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du phu de bang trong HTML**</title>**

**</head>**

**<body>**

**<table** border="1" width="100%"**>**

**<caption>**Vi du phu de bang**</caption>**

**<tr>**

**<td>**row 1, column 1**</td><td>**row 1, columnn 2**</td>**

**</tr>**

**<tr>**

**<td>**row 2, column 1**</td><td>**row 2, columnn 2**</td>**

**</tr>**

**</table>**

**</body>**

**</html>**

Đoạn mã này sẽ tạo ra kết quả sau:

|  |  |
| --- | --- |
| Vi du phu de bang | |
| row 1, column 1 | row 1, columnn 2 |
| row 2, column 1 | row 2, columnn 2 |

Đầu, thân và chân bảng trong HTML:

Bảng có thể được chia làm 3 phần: một tiêu đề, phần thân chứa dữ liệu và một phần cuối bảng chứa dòng tổng kết. Đầu và chân bảng thì tương tự như đầu trang và chân trang trong các tài liệu văn bản, trong khi phần thân là nội dung chính được hiển thị trong bảng.

Dưới đây là 3 phần tử riêng rẽ trong một bảng:

* **<thead> -** Tạo một tiêu đề bảng riêng biệt.
* **<tbody> -** Chỉ nội dung chính của bảng.
* **<tfoot> -** Tạo một phần chân bảng riêng biệt.

Một bảng có thể chứa nhiều phần tử <tbody> để chỉ các vùng, nhóm của dữ liệu. Tuy nhiên điều lưu ý là các thẻ <thead> và <tfoot> sẽ xuất hiện trước các thẻ <tbody>

Ví dụ

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du bang trong HTML**</title>**

**</head>**

**<body>**

**<table** border="1" width="100%"**>**

**<thead>**

**<tr>**

**<td** colspan="4"**>**Day la phan tieu de bang**</td>**

**</tr>**

**</thead>**

**<tfoot>**

**<tr>**

**<td** colspan="4"**>**Day la phan chan bang**</td>**

**</tr>**

**</tfoot>**

**<tbody>**

**<tr>**

**<td>**Cell 1**</td>**

**<td>**Cell 2**</td>**

**<td>**Cell 3**</td>**

**<td>**Cell 4**</td>**

**</tr>**

**</tbody>**

**</table>**

**</body>**

**</html>**

Kết quả là:

|  |  |  |  |
| --- | --- | --- | --- |
| Day la phan tieu de bang | | | |
| Day la phan chan bang | | | |
| Cell 1 | Cell 2 | Cell 3 | Cell 4 |

Lồng bảng vào trong bảng trong HTML:

Bạn có thể lồng một bảng vào trong một bảng khác. Bạn có thể sử dụng hầu hết tất cả các thẻ bên trong thẻ <td>.

Ví dụ

Ví dụ dưới đây sử dụng một bảng khác và các thẻ khác bên trong một ô bảng.

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du bang trong HTML**</title>**

**</head>**

**<body>**

**<table** border="1" width="100%"**>**

**<tr>**

**<td>**

**<table** border="1" width="100%"**>**

**<tr>**

**<th>**Ten nhan vien**</th>**

**<th>**Luong**</th>**

**</tr>**

**<tr>**

**<td>**Minh Chinh**</td>**

**<td>**5000**</td>**

**</tr>**

**<tr>**

**<td>**Duy Manh**</td>**

**<td>**7000**</td>**

**</tr>**

**</table>**

**</td>**

**</tr>**

**</table>**

**</body>**

**</html>**

Nó sẽ cho kết quả như sau:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Ten nhan vien** | **Luong** | | Minh Chinh | 5000 | | Duy Manh | 7000 | |

### HTML Lists

**Danh sách HTML không có thứ tự**

Một danh sách không có thứ tự bắt đầu với thẻ. Mỗi mục danh sách bắt đầu với thẻ. <ul><li>

Các mục danh sách sẽ được đánh dấu bằng đạn (vòng tròn nhỏ màu đen) theo mặc định:

Thí dụ

<ul>

  <li>Coffee</li>

  <li>Tea</li>

  <li>Milk</li>

</ul>

Danh sách không có thứ tự trong HTML - Thẻ ul trong HTML

Là loại danh sách mà chưa được sắp qua thứ tự. Để tạo loại danh sách này bạn sử dụng thẻ <ul>. Mỗi mục trong danh sách được kết nối với một bullet.

Ví dụ

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du danh sach chua qua sap xep**</title>**

**</head>**

**<body>**

**<ul>**

**<li>**Beetroot**</li>**

**<li>**Ginger**</li>**

**<li>**Potato**</li>**

**<li>**Radish**</li>**

**</ul>**

**</body>**

**</html>**

Kết quả là như sau:

* Beetroot
* Ginger
* Potato
* Radish

**Thuộc tính type trong HTML**

Bạn có thể sử dụng thuộc tính type cho thẻ <ul> để xác định kiểu của bullet mà bạn thích. Theo mặc định nó có hình dạng chiếc đĩa tròn (disc). Có các tùy chọn kiểu cho bạn sử dụng:

<ul type="square">

<ul type="disc">

<ul type="circle">

Ví dụ

Ví dụ sau chúng tôi sử dụng <ul type="square">

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du danh sach chua qua sap xep**</title>**

**</head>**

**<body>**

**<ul** type="square"**>**

**<li>**Beetroot**</li>**

**<li>**Ginger**</li>**

**<li>**Potato**</li>**

**<li>**Radish**</li>**

**</ul>**

**</body>**

**</html>**

Kết quả là:

* Beetroot
* Ginger
* Potato

Ví dụ

sử dụng <ul type="disc"> trong thư viện HTML:

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du danh sach chua qua sap xep**</title>**

**</head>**

**<body>**

**<ul** type="disc"**>**

**<li>**Beetroot**</li>**

**<li>**Ginger**</li>**

**<li>**Potato**</li>**

**<li>**Radish**</li>**

**</ul>**

**</body>**

**</html>**

Kết quả hiển thị là:

* Beetroot
* Ginger
* Potato
* Radish

Ví dụsử dụng <ul type="circle"> trong thư viện HTML:

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du danh sach chua qua sap xep**</title>**

**</head>**

**<body>**

**<ul** type="circle"**>**

**<li>**Beetroot**</li>**

**<li>**Ginger**</li>**

**<li>**Potato**</li>**

**<li>**Radish**</li>**

**</ul>**

**</body>**

**</html>**

Nó sẽ tạo ra kết quả sau:

* Beetroot
* Ginger
* Potato
* Radish

**Danh sách đã qua sắp xếp trong HTML**

Nếu bạn được yêu cầu đặt các mục trong danh sách theo thứ tự số thay vì sử dụng các bullet thì loại danh sách đã qua sắp xếp sẽ được sử dụng. Danh sách này được tạo bởi thẻ <ol>. Thứ tự số bắt đầu từ 1 và lượng gia thêm một cho các mục tiếp theo với thẻ <li>.

Ví dụ

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du danh sach da qua sap xep**</title>**

**</head>**

**<body>**

**<ol>**

**<li>**Beetroot**</li>**

**<li>**Ginger**</li>**

**<li>**Potato**</li>**

**<li>**Radish**</li>**

**</ol>**

**</body>**

**</html>**

Nó sẽ tạo ra kết quả sau:

1. Beetroot
2. Ginger
3. Potato
4. Radish

**Thuộc tính type trong HTML**

Bạn có thể sử dụng thuộc tính type cho thẻ <ol> để xác định kiểu của loại thứ tự số mà bạn thích. Theo mặc định, nó là một số. Dưới đây là các tùy chọn có thể:

<ol type="1"> - Gia tri so mac dinh.

<ol type="I"> - Gia tri so La Ma dang chu hoa.

<ol type="i"> - Gia tri so La Ma dang chu thuong.

<ol type="a"> - Chu cai thuong.

<ol type="A"> - Chu cai hoa.

Ví dụ

Ví dụ sau chúng tôi sử dụng <ol type="1">

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du danh sach da qua sap xep**</title>**

**</head>**

**<body>**

**<ol** type="1"**>**

**<li>**Beetroot**</li>**

**<li>**Ginger**</li>**

**<li>**Potato**</li>**

**<li>**Radish**</li>**

**</ol>**

**</body>**

**</html>**

Kết quả là:

1. Beetroot
2. Ginger
3. Potato
4. Radish

Ví dụsử dụng <ol type="I">

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du danh sach da qua sap xep**</title>**

**</head>**

**<body>**

**<ol** type="I"**>**

**<li>**Beetroot**</li>**

**<li>**Ginger**</li>**

**<li>**Potato**</li>**

**<li>**Radish**</li>**

**</ol>**

**</body>**

**</html>**

Kết quả khi chạy đoạn code trên là:

I. Beetroot

II. Ginger

III. Potato

IV. Radish

Ví dụsử dụng <ol type="i">

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du danh sach da qua sap xep**</title>**

**</head>**

**<body>**

**<ol** type="i"**>**

**<li>**Beetroot**</li>**

**<li>**Ginger**</li>**

**<li>**Potato**</li>**

**<li>**Radish**</li>**

**</ol>**

**</body>**

**</html>**

Khi chạy đoạn code trên, nó sẽ tạo ra kết quả sau:

i. Beetroot

ii. Ginger

iii. Potato

iv. Radish

Ví dụ

Ví dụ này chúng tôi sử dụng <ol type="A">

<html><!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du danh sach da qua sap xep**</title>**

**</head>**

**<body>**

**<ol** type="A"**>**

**<li>**Beetroot**</li>**

**<li>**Ginger**</li>**

**<li>**Potato**</li>**

**<li>**Radish**</li>**

**</ol>**

**</body>**

**</html>**

Kết quả là:

A. Beetroot

B. Ginger

C. Potato

D. Radish

Ví dụsử dụng <ol type="a">

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du danh sach da qua sap xep**</title>**

**</head>**

**<body>**

**<ol** type="a"**>**

**<li>**Beetroot**</li>**

**<li>**Ginger**</li>**

**<li>**Potato**</li>**

**<li>**Radish**</li>**

**</ol>**

**</body>**

**</html>**

Nó tạo ra danh sách dạng sau:

a. Beetroot

b. Ginger

c. Potato

d. Radish

**Thuộc tính start trong HTML**

Bạn có thể sử dụng thuộc tính start cho thẻ <ol> để xác định điểm bắt đầu của dãy số bạn muốn. Dưới đây là các tùy chọn có thể:

<ol type="1" start="4"> - Day so bat dau tu 4.

<ol type="I" start="4"> - Day so bat dau tu IV.

<ol type="i" start="4"> - Day so bat dau tu iv.

<ol type="a" start="4"> - Day chu cai bat dau tu d.

<ol type="A" start="4"> - Day chu cai bat dau tu D.

Ví dụsử dụng <ol type="i" start="4" >

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du danh sach da qua sap xep**</title>**

**</head>**

**<body>**

**<ol** type="i" start="4"**>**

**<li>**Beetroot**</li>**

**<li>**Ginger**</li>**

**<li>**Potato**</li>**

**<li>**Radish**</li>**

**</ol>**

**</body>**

**</html>**

Nó sẽ tạo kết quả:

iv. Beetroot

v. Ginger

vi. Potato

vii. Radish

**Danh sách định nghĩa trong HTML - Thẻ dl trong HTML**

HTML và XHTML hỗ trợ một kiểu danh sách mà được gọi là Definition list - Danh sách định nghĩa, là nơi mà các mục được liệt kê dưới dạng giống một từ điển hoặc một quyển bách khoa toàn thư. Danh sách này là một cách tuyệt vời để hiển thị một bảng danh sách, bảng chú giải của các mục dữ liệu.

Danh sách định nghĩa sử dụng 3 thẻ theo sau:

<dl> - Xác định phần bắt đầu của danh sách

<dt> - Một mục

<dd> - Định nghĩa của mục

</dl> - Xác định phần kết thúc của danh sách

Ví dụ

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du the dl trong HTML**</title>**

**</head>**

**<body>**

**<dl>**

**<dt><b>**HTML**</b></dt>**

**<dd>**La viet tat cua Hyper Text Markup Language**</dd>**

**<dt><b>**HTTP**</b></dt>**

**<dd>**La viet tat cua Hyper Text Transfer Protocol**</dd>**

**</dl>**

**</body>**

**</html>**

Kết quả khi chạy đoạn code trên là:

HTML

La viet tat cua Hyper Text Markup Language

HTTP

La viet tat cua Hyper Text Transfer Protocol

### HTML Blocks

Tất cả các phần tử trong HTML có thể được chia thành hai loại: các phần tử cấp khối và các phần tử nội tuyến.

Phần tử cấp khối trong HTML

Phần tử cấp khối thường bắt đầu và kết thúc với một dòng mới khi hiển thị trong trình duyệt. Ví dụ cho các phần tử này là <p>, <h1>, <h2>, <h3>, <h4>, <h5>, <h6>, <ul>, <ol>, <dl>, <pre>, <hr />, <blockquote>, và <address>. Tất cả chúng bắt đầu bởi một dòng mới, và bất cứ cái gì theo sau chúng, đều xuất hiện trên dòng mới này.

Phần tử nội tuyến trong HTML

Các phần tử nội tuyến xuất hiện trong một câu và không cần phải cách dòng. Ví dụ cho các phần tử này là <b>, <i>, <u>, <em>, <strong>, <sup>, <sub>, <big>, <small>, <li>, <ins>, <del>, <code>, <cite>, <dfn>, <kbd>, và <var>.

Tạo nhóm các phần tử HTML trong HTML

Có 2 thẻ quan trọng mà chúng ta sử dụng thường xuyên để tạo nhóm các thẻ HTML khác là (i) <div> và (ii) <span>.

Thẻ <div> trong HTML

Đây là một thẻ cấp khối rất quan trọng đóng vai trò lớn trong việc tạo nhóm các thẻ HTML khác và áp dụng CSS trong nhóm các phần tử. Ngay trong hiện tại thì thẻ <div> có thể được sử dụng để tạo cách bố trí (layout) cho web, tại đây chúng ta xác định các phần khác nhau (trái, phải, trên,…) của trang bằng cách sử dụng thẻ này. Thẻ này không cung cấp bất cứ sự thay đổi về thị giác trên khối nhưng nó có nhiều ý nghĩa hơn khi được sử dụng cùng với CSS.

Dưới đây là một ví dụ đơn giản về việc sử dụng thẻ <div>. Chúng ta sẽ học về CSS trong một chương riêng, nhưng tại đây chúng ta sử dụng nó để thấy sự hữu ích khi dùng thẻ <div>:

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du the div trong HTML**</title>**

**</head>**

**<body>**

<!-- Nhom so 1 -->

**<div** style="color:red"**>**

**<h4>**Day la nhom dau tien**</h4>**

**<p>**Danh sach cac loai rau**</p>**

**<ul>**

**<li>**Beetroot**</li>**

**<li>**Ginger**</li>**

**<li>**Potato**</li>**

**<li>**Radish**</li>**

**</ul>**

**</div>**

<!-- Nhom so 2 -->

**<div** style="color:green"**>**

**<h4>**Day la nhom thu hai**</h4>**

**<p>**Danh sach cac loai trai cay**</p>**

**<ul>**

**<li>**Apple**</li>**

**<li>**Banana**</li>**

**<li>**Mango**</li>**

**<li>**Strawberry**</li>**

**</ul>**

**</div>**

**</body>**

</html>

**Thẻ <span> trong HTML**

Thẻ <span> là một phần tử nội tuyến và có thể được sử dụng để tạo nhóm nội tuyến trong tài liệu HTML. Thẻ này cũng không cung cấp bất cứ sự thay đổi về thị giác trên khối, nhưng nó có nhiều ý nghĩa hơn khi sử dụng cùng với CSS.

Sự khác nhau giữa thẻ <span> và thẻ <div> là thẻ <span> được sử dụng với các phần tử nội tuyến trong khi thẻ <div> được sử dụng với các phần tử cấp khối.

Ví dụ

Dưới đây là ví dụ đơn giản về thẻ <span>. Chúng ta sẽ học CSS ở một chương riêng nhưng hiện tại chúng ta tìm hiểu nó để thấy sự hữu ích của thẻ <span> này:

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du the span trong HTML**</title>**

**</head>**

**<body>**

**<p>**Day la **<span** style="color:red"**>**mau do**</span>** va day la **<span** style="color:green"**>**mau xanh**</span></p>**

**</body>**

**</html>**

### HTML Classes

Trong HTML, thuộc tính class dùng để xác định một hoặc nhiều tên lớp cho phần tử HTML.

Vì vậy, tất cả các phần tử HTML cùng thuộc tính class sẽ có kiểu định nghĩa giống nhau.

Dưới đây chúng ta có ba phần tử <div> cùng trỏ đến một class.

Ví dụ:

<!DOCTYPE html>  
<html>  
<head>  
<style>  
.cities {  
  background-color: black;  
  color: white;  
  margin: 20px;  
  padding: 20px;  
}  
</style>  
</head>  
<body>  
  
<div class="cities">  
  <h2>London</h2>  
  <p>London is the capital of England.</p>  
</div>  
  
<div class="cities">  
  <h2>Paris</h2>  
  <p>Paris is the capital of France.</p>  
</div>  
  
<div class="cities">  
  <h2>Tokyo</h2>  
  <p>Tokyo is the capital of Japan.</p>  
</div>  
  
</body>  
</html>

Kết quả :



* · Sử dụng thuộc tính trong các thẻ Inline

Thuộc tính HTML class có thể được sử dụng trong các thẻ Inline

Ví dụ:

<!DOCTYPE html>  
<html>  
<head>  
<style>  
span.note {  
  font-size: 120%;  
  color: red;  
}  
</style>  
</head>  
<body>  
  
<h1>My <span class="note">Important</span> Heading</h1>  
<p>This is some <span class="note">important</span> text.</p>  
  
</body>  
</html>

Lưu ý :

* Thuộc tính class có thể dùng với bất kỳ phần tử HTML nào.
* Tên lớp có phân biệt giữa chữ viết hoa và viết thường.
* Chọn phần tử HTML với một class đặt biệt.

Trong CSS, để chọn phần tử với một class đặc biệt, ta viết dấu chấm (.)tên thuộc tính.

Ví dụ :

Sử dụng CSS để định dạng tất cả các thuộc tính có tên class “city”

<style>  
.city {  
  background-color: tomato;  
  color: white;  
  padding: 10px;  
}  
</style>  
  
<h2 class**="city"**>London</h2>  
<p>London is the capital of England.</p>  
  
<h2 class**="city"**>Paris</h2>  
<p>Paris is the capital of France.</p>  
  
<h2 class**="city"**>Tokyo</h2>  
<p>Tokyo is the capital of Japan.</p>

Kết quả :



* · Sử dụng nhiều class

Mỗi phần tử HTML có thể có nhiều hơn một class, mỗi tên class phải được ngăn cách bởi một dấu cách.

Ví dụ :

Định dạng các phần tử HTML với class “city”, có thể định dạng phần tử HTML với class “main”.

<h2 class="city main">London</h2>  
<h2 class="city">Paris</h2>  
<h2 class="city">Tokyo</h2>

Trong ví dụ trên, phần tử <h2> được định dạng bởi cả 2 class “city” và “main”.

· Các thẻ khác nhau có thể sử dụng chung một class

Các thẻ khác nhau, như <h2> và <p>, có thể có cùng tên class và qua đó được định dạng cùng một kiểu.

Ví dụ :

<h2 class="city">Paris</h2>  
<p class="city">Paris is the capital of France</p>

* · Sử dụng thuộc tính class trong JavaScript

Tên class cũng có thể được sử dụng bởi các đoạn JavaScript để thực hiện nhiệm vụ nhất định cho các phần tử với tên class được chỉ định.

JavaScript có thể truy cập đến phần tử với một tên class được chỉ định bằng cách sử dụng phương thức getElenmentsByClassName() :

Ví dụ :

Khi người dùng click vào một button, ẩn tất cả các phần tử với tên class “city”

<script>  
function myFunction() {  
  var x = document.**getElementsByClassName**(**"city"**);  
  for (var i = 0; i < x.length; i++) {  
    x[i].style.display = "none";  
  }  
}  
</script>

· Bài tập

* Tạo một class với tên “special”.
* Thêm các thuộc tính màu sắc với giá trị “blue” bên trong class “special”.

### HTML Id

* · Sử dụng thuộc tính id
* Thuộc tính id dùng để đặt tên cho phần tử trong HTML, tên này phải là duy nhất.
* Giá trị id có thể được sử dụng bởi CSS và JavaScript để thực hiện nhiệm vụ nhất định cho các phần tử có giá trị id cụ thể.
* Trong CSS, để chọn một phần tử với id cụ thể, ta viết #id của phần tử.

Ví dụ:

Sử dụng CSS để định dạng một phần tử với id “myHeader”

<style>  
#myHeader {  
  background-color: lightblue;  
  color: black;  
  padding: 40px;  
  text-align: center;  
}  
</style>  
  
<h1 id="myHeader">My Header</h1>

Lưu ý:

* Thuộc tính id có thể được sử dụng cho bất kì phần tử HTML nào.
* Giá trị id là duy nhất.
* Các giá trị id phải chứa ít nhất một ký tự, và không được chứa khoảng trắng.
* Sự khác biệt giữa Class và ID

Một phần tử HTML chỉ có thể có một id duy nhất, trong khi đó một class có thể sử dụng bởi nhiều phần tử.

Ví dụ:

<style>  
/\* Style the element with the id "myHeader" \*/  
#myHeader {  
  background-color: lightblue;  
  color: black;  
  padding: 40px;  
  text-align: center;  
}  
/\* Style all elements with the class name "city" \*/  
.city {  
  background-color: tomato;  
  color: white;  
  padding: 10px;  
}  
</style>  
<!-- A unique element -->  
<h1 id="myHeader">My Cities</h1>  
<!-- Multiple similar elements -->  
<h2 class="city">London</h2>  
<p>London is the capital of England.</p>  
<h2 class="city">Paris</h2>  
<p>Paris is the capital of France.</p>  
<h2 class="city">Tokyo</h2>  
<p>Tokyo is the capital of Japan.</p>

* · Bookmarks với ID và Links
* HTML Bookmark được sử dụng để cho phép người đọc có thể chuyển đến các phần cụ thể của một trang Web.
* Bookmarks sẽ rất hữu ích, nếu như trang Web của bạn quá dài.
* Để tạo một Bookmark, trước tiên bạn phải tạo Bookmark, sau đó thêm một đường link để dẫn đến nó.
* Khi click vào đường link, trang Web sẽ di chuyển đến vị trí Bookmark.

Ví dụ:

* Đầu tiên, tạo một Bookmark với thuộc tính id:

<h2 id="C4">Chapter 4</h2>

* Sau đó, thêm đường link trỏ tới Bookmark (“chuyển đến Chapter 4”), từ bên trong cùng một trang.

<a href="#C4">Jump to Chapter 4</a>

- Hoặc , thêm đường link trỏ tới Bookmark (“chuyển đến Chapter 4”), từ một trang khác:

<a href="#C4">Jump to Chapter 4</a>

* · Sử dụng thuộc tính id trong JavaScript

JavaScript có thể truy cập vào một phần tử với id xác định bằng cách sử dụng phương thức getElenmentById() :

Ví dụ: Sử dụng thuộc tính id để thao tác văn bản với JavaScript :

<script>  
function displayResult() {  
  document.getElementById("myHeader").innerHTML = "Have a nice day!";  
}  
</script>

Bài tập :

Thêm các thuộc tính HTML để làm phần tử H1 có màu đỏ.

<!DOCTYPE html>  
<html>  
<head>  
<style>  
#myheader {color:red;}  
</style>  
</head>  
<body>  
  
<h1 >My Home Page</h1>  
  
</body>

</html>

### HTML Forms

#### **HTML Forms**

* Ví dụ HTML Form



* Phần tử <form>

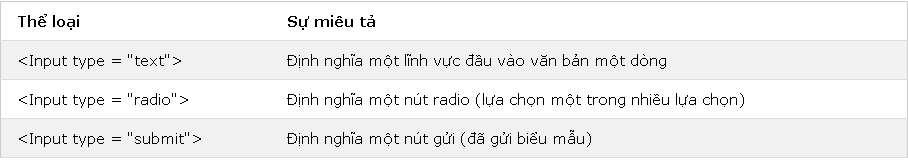
Phần tử HTML <form> định nghĩa một form được dùng để thu thập dữ liệu đầu vào của người dùng.

<form>  
.  
*form elements*  
.  
</form>

Một HTML form bao gồm các phần tử form.

Phần tử Form có thể nhận nhiều kiểu thuộc tính đầu vào, như văn bản, checkbox, nút radio, nút button,...

* Phần tử <input>
* Phần tử <input> là yếu tố hình thức quan trọng nhất.
* Các ohanaf tử <input> có thể được hiển thị theo nhiều cách, tùy thuộc vào loại thuộc tính.
* Dưới đây là một số ví dụ:



* Thuộc tính text Input

<input type= “text”> xác định một dòng dữ liệu đầu vào kiểu text.

Ví dụ :

<form>  
  First name:<br>  
  <input type="text" name="firstname"><br>  
  Last name:<br>  
  <input type="text" name="lastname">  
</form>

Kết quả:



Chú ý : Hình thức cùa form không thể nhìn thấy được. Độ rộng một trường văn bản là tối đa 20 ký tự.

* Radio Button Input

<input type = “radio”> xác định một nút radio.

Nút Radio cho người dùng chọn một trong số giới hạn lựa chọn.

Ví dụ :

<form>  
  <input type="radio" name="gender" value="male" checked> Male<br>  
  <input type="radio" name="gender" value="female"> Female<br>  
  <input type="radio" name="gender" value="other"> Other  
</form>

Kết quả :



* The Submit Button

<input type = “submit”> xác định một nút Submiting gửi dữ liệu trong form đến form-handler.

Form-handler thường là một máy chủ với một kịch bản để xử lý dữ liệu đầu vào.

Form-handler được quy định ở thuộc tính action của form.

Ví dụ:

<form action="/action\_page.php">  
  First name:<br>  
  <input type="text" name="firstname" value="Mickey"><br>  
  Last name:<br>  
  <input type="text" name="lastname" value="Mouse"><br><br>  
  <input type="submit" value="Submit">  
</form>

Đoạn mã HTML trên sẽ được hiển thị trên trình duyệt như sau :



* Thuộc tính Action
* Thuộc tính action xác định hành động sẽ được thực hiện khi form được submit.
* Thông thường, dữ liệu của form sẽ được gửi đến một trang Web trên máy chủ khi người dùng nhấp vào nút Submit.
* Trong ví dụ ở trên, các dữ liệu của form sẽ được gửi đến một trang trên máy chủ gọi là “/action\_page.php”. Trang này chứa một server-side script để xử lý các dữ liệu của form :

<form action="/action\_page.php">

* Nếu thuộc tính action được bỏ qua , các hành động được thiết lập để trang hiện tại.
* Thuộc tính Target
* Các thuộc tính Target xác định nếu kết quả submitted sẽ mở trong một tab trình duyệt mới, một khung hình , hoặc trong cửa sổ hiện hành.
* Giá trị mặc định là “\_self”, có nghĩa là form được đệ trình trong cửa sổ hiện hành.
* Để kết quả của form mở trong một tab trình duyệt mới, sử dụng giá trị “\_blank”.
* Ví dụ :

<form action="/action\_page.php" target**="\_blank"**>

* Giá trị pháp lý khác là “\_parent”, “\_top”, hoặc một tên đại diện cho một ifame.
* Thuộc tính Method

Thuộc tính Method quy định cụ thể phương thức HTTP ( GET hoặc POST ) được sử dụng khi gửi dữ liệu của form.

Ví dụ :

<form action="/action\_page.php" method**="get"**>

Hoặc :

<form action="/action\_page.php" method**="post"**>

* Khi sử dụng GET?
* Phương thức mặc định khi gửi dữ liệu của form là GET.
* Tuy nhiên, khi GET được sử dụng, dữ liệu đã gửi của form sẽ có thể nhìn thấy trong lĩnh vực địa chỉ trang.

/action\_page.php?firstname=Mickey&lastname=Mouse

* Chú ý khi dùng GET :
* Gắn thêm form-data vào URL trong tên/cặp giá trị.
* Chiều dài của một URL được giới hạn (khoảng 3000 ký tự)
* Không bao giờ sử dụng GET để gửi dữ liệu nhạy cảm! (Sẽ được hiển thị trong URL)
* Hữu ích cho hình thức đệ trình mà người dùng muốn đánh dấu kết quả.
* GET phù hợp hơn cho dữ liệu không an toàn, giống như chuỗi truy vấn trong Google.
* Khi sử dụng POST?

Luôn luôn sử dụng POST nếu dữ liệu dạng chứa thông tin nhạy cảm hoặc cá nhân. Các phương thức POST không hiển thị dữ liệu trong lĩnh vực địa chie trang.

Chú ý khi sử dụng POST:

* POST không có giới hạn kích thước, và có thể sử dụng để gửi một lượng lớn dữ liệu.
* Mẫu đệ trình với POST không thể đánh dấu.
* Thuộc tính Name

Mỗi trường đầu vào cần phải có một thuộc tính name để gửi dữ liệu.

Nếu thuộc tính name bị bở qua, dữ liệu ở trường đầu vào sẽ không được gửi hết.

Ví dụ này sẽ chỉ submit trường đầu vào “Last name” :

<form action="/action\_page.php">  
  First name:<br>  
  <input type="text" value="Mickey"><br>  
  Last name:<br>  
  <input type="text" name="lastname" value="Mouse"><br><br>  
  <input type="submit" value="Submit">  
</form>

* Nhóm dữ liệu của form với <fieldset>

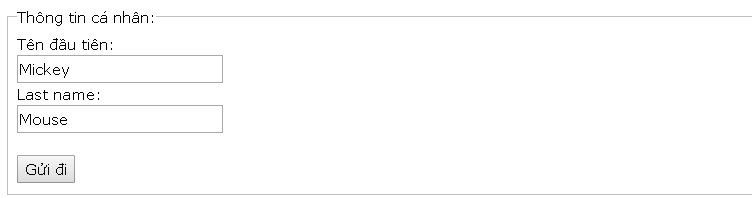
Phần tử <fieldset> được dùng để nhóm dữ liệu liên quan trong một biểu mẫu.

Phần tử <legend> định nghĩa một chú thích cho phần tử <fieldset>.

Ví dụ :

<form action="/action\_page.php">  
  <fieldset>  
    <legend>Personal information:</legend>  
    First name:<br>  
    <input type="text" name="firstname" value="Mickey"><br>  
    Last name:<br>  
    <input type="text" name="lastname" value="Mouse"><br><br>  
    <input type="submit" value="Submit">  
  </fieldset>  
</form>

Mã code trên sẽ được hiển thị trong trình duyệt như sau :



* Bài tập

Trong form dưới đây, thêm một trường dữ liệu đầu vào kiểu “button” và giá trị “OK”.

<form>  
<>  
</form>

#### **HTML Form Elements**

* Phần tử <input>

Phần tử quan trọng nhất của form là <input>

Phần tử <input> có thể được hiển thị bằng nhiều cách, phụ thuộc vào thuộc tính type.

Ví dụ :

<input name="firstname" type="text">

Nếu thuộc tính type được bỏ qua, trường dữ liệu đầu vào sẽ lấy type mặc định là “text”.

* Phần tử <select>

Phần tử <select> định nghĩa một danh sách thả xuống.

Ví dụ :

<select name="cars">  
  <option value="volvo">Volvo</option>  
  <option value="saab">Saab</option>  
  <option value="fiat">Fiat</option>  
  <option value="audi">Audi</option>  
</select>

Phần tử <option> định nghĩa một tùy chọn, có thể được lựa chọn.

Mặc định, mục đầu tiên của danh sách thả xuống được chọn.

Để xác định một tùy chọn được chọn sẵn, thêm thuộc tính select để lựa chọn :

<option value="fiat" selected>Fiat</option>

* Các giá trị có thể nhìn thấy

Dùng thuộc tính size để xác định những giá trị có thể nhìn thấy :

<select name="cars" size**="3"**>  
  <option value="volvo">Volvo</option>  
  <option value="saab">Saab</option>  
  <option value="fiat">Fiat</option>  
  <option value="audi">Audi</option>  
</select>

* Cho phép nhiều lựa chọn

Sử dụng thuộc tính multiple để cho phép người dùng chọn nhiều hơn một giá trị:

<select name="cars" size="4" multiple>  
  <option value="volvo">Volvo</option>  
  <option value="saab">Saab</option>  
  <option value="fiat">Fiat</option>  
  <option value="audi">Audi</option>  
</select>

* Phần tử <textarea>

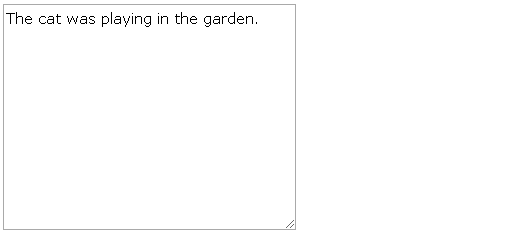
Phần tử <textarea> xác định một trường nhập nhiều dòng (a text area) :

<textarea name="message" rows="10" cols="30">  
The cat was playing in the garden.  
</textarea>

Thuộc tính rows xác định số dòng có thể nhìn thấy trong vùng văn bản.

Thuộc tính cols xác định chiều rộng hữu hình của vùng văn bản.

Đoạn mã HTML bên trên được hiển thị trên trình duyệt :



Bạn có thể xác định kích cỡ của vùng văn bản bằng cách sử dụng CSS :

<textarea name="message" style="width:200px; height:600px;">  
The cat was playing in the garden.  
</textarea>

* Phần tử <button>

Phần tử <button> xác định một nút có thể nhấp vào :

<button type="button" onclick="alert('Hello World!')">Click Me!</button>

Chú ý : Luôn luôn xác định thuộc tính type cho phần tử button. Các trình duyệt khác nhau có thể sử dụng các kiểu mặc định khác nhau cho phần tử button.

* Phần tử HTML5 Form
* HTML5 thêm các phần tử form sau đây :

<datalist>

<output>

* Chú ý : Trình duyệt không hiển thị những phần tử chưa biết. Những phần tử mới không được hỗ trợ trong các trình duyệt cũ sẽ không “destroy” trang web của bạn.
* Phần tử HTML5 <datalist>
* Phần tử <datalist> xác định một danh sách các tùy chọn được xác định trước cho một phần tử <input>.
* Người dùng sẽ thấy một danh sách thả xuống trong những lựa chọn được xác định trước khi họ nhập dữ liệu.
* Các thuộc tính list của phần tử <input>, phải chỉ đến thuộc tính id của phần tử <datalist>.
* Ví dụ :

<form action="/action\_page.php">  
  <input list="browsers">  
  <datalist id="browsers">  
    <option value="Internet Explorer">  
    <option value="Firefox">  
    <option value="Chrome">  
    <option value="Opera">  
    <option value="Safari">  
  </datalist>  
</form>

* Phần tử HTML5 <output>

Phần tử <output> đại diện cho kết quả của một phép tính ( giống như được thực hiện bởi một kịch bản) :

Ví dụ : Thực hiện tính toán và hiển thị kết quả trong một phần tử <output>

<form action="/action\_page.php"  
  oninput="x.value=parseInt(a.value)+parseInt(b.value)">  
  0  
  <input type="range"  id="a" name="a" value="50">  
  100 +  
  <input type="number" id="b" name="b" value="50">  
  =  
  <output name="x" for="a b"></output>  
  <br><br>  
  <input type="submit">  
</form>

* Bài tập

Trong mẫu dưới đây, thêm một danh sách thả xuống rỗng với tên “cars”.

<form action="/action\_page.php">  
<>  
</>  
</form>

#### **HTML Input Types**

Dưới đây là các loại đầu vào khác nhau mà bạn có thể sử dụng trong HTML:

* <input type="button">
* <input type="checkbox">
* <input type="color">
* <input type="date">
* <input type="datetime-local">
* <input type="email">
* <input type="file">
* <input type="hidden">
* <input type="image">
* <input type="month">
* <input type="number">
* <input type="password">
* <input type="radio">
* <input type="range">
* <input type="reset">
* <input type="search">
* <input type="submit">
* <input type="tel">
* <input type="text">
* <input type="time">
* <input type="url">
* <input type="week">
* Input type Text

<input type="text">xác định một dòng trường nhập văn bản.

Ví dụ

<form>  
  First name:<br>  
  <input type="text" name="firstname"><br>  
  Last name:<br>  
  <input type="text" name="lastname">  
</form>

Đoạn code HTML trên sẽ được hiển thị trên trình duyệt:



* Input type Password

<input type="password">xác định một trường nhập mật khẩu.

Ví dụ:

<form>  
  User name:<br>  
  <input type="text" name="username"><br>  
  User password:<br>  
  <input type="password" name="psw">  
</form>

Đoạn code HTML trên sẽ được hiển thị trên trình duyệt:



Đối tượng trong trường mật khẩu sẽ được che (hiển thị như dấu sao hoặc hình tròn).

* Input Type Submit

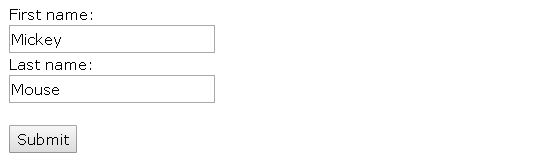
<input type="submit">định nghĩa một nút để gửi dữ liệu trong form đến form-handler.

Form-handler là một loại máy chủ với một kịch bản để xử lý dữ liệu đầu vào.

Form-handler được quy định bởi thuộc tính action của Form:

<form action="/action\_page.php">  
  First name:<br>  
  <input type="text" name="firstname" value="Mickey"><br>  
  Last name:<br>  
  <input type="text" name="lastname" value="Mouse"><br><br>  
  <input type="submit" value="Submit">  
</form>

Đoạn code HTML trên sẽ được hiển thị trên trình duyệt:



Nếu bạn bở qua giá trị thuộc tính của nút submit, nút sẽ nhận được một văn bản mặc định.

<form action="/action\_page.php">  
  First name:<br>  
  <input type="text" name="firstname" value="Mickey"><br>  
  Last name:<br>  
  <input type="text" name="lastname" value="Mouse"><br><br>  
  <input type="submit">  
</form>

* Input Type Reset

<input type="reset">định nghĩa một nút reset mà sẽ đặt lại toàn bộ giá trị của form thành giá trị mặc định.

Ví dụ :

<form action="/action\_page.php">  
  First name:<br>  
  <input type="text" name="firstname" value="Mickey"><br>  
  Last name:<br>  
  <input type="text" name="lastname" value="Mouse"><br><br>  
  <input type="submit" value="Submit">  
  <input type="reset">  
</form>

Đoạn code HTML trên sẽ được hiển thị trên trình duyệt:



Nếu bạn thay đổi các giá trị đầu vào và sau đó nhấp vào nút “Reset”, dữ liệu trong form sẽ được đặt lại về giá trị mặc định.

* Input Type Radio

<input type="radio">định nghĩa một nút radio.

Nút Radio để người dùng chọn chỉ một trong một số giới hạn các lựa chọn :

<form>  
  <input type="radio" name="gender" value="male" checked> Male<br>  
  <input type="radio" name="gender" value="female"> Female<br>  
  <input type="radio" name="gender" value="other"> Other  
</form>

Đoạn code HTML trên sẽ được hiển thị trên trình duyệt:



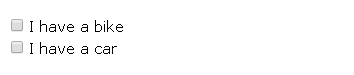
* Input Type Checkbox

<input type="checkbox">định nghĩa một checkbox.

Checkbox cho phép người dùng chọn không hay nhiều lựa chọn trong số giới hạn các lựa chọn :

<form>  
  <input type="checkbox" name="vehicle1" value="Bike"> I have a bike<br>  
  <input type="checkbox" name="vehicle2" value="Car"> I have a car  
</form>

Đoạn code HTML trên sẽ được hiển thị trên trình duyệt:



* Input Type Button

<input type="button">định nghĩa một button.

<input type="button" onclick="alert('Hello World!')" value="Click Me!">

Đoạn code HTML trên sẽ được hiển thị trên trình duyệt:



HTML5 thêm một số kiểu đầu vào mới :

* color
* date
* datetime-local
* email
* month
* number
* range
* search
* tel
* time
* url
* week

Các kiểu đầu vào mới không được hỗ trợ bởi các trình duyệt cũ, sẽ được xử lý như <input type="text">

* Input Type Color

<input type="color">được sử dụng cho trường đầu vào mà cần chứa một màu.

Tùy thuộc vào sự hỗ trợ của trình duyệt, một bảng chọn màu có thể được hiển thị trong trường đầu vào.

<form>  
  Select your favorite color:  
  <input type="color" name="favcolor">  
</form>

* Input Type Date

<input type="date">được dùng cho trường đầu vào mà chứa một ngày.

Tùy thuộc vào sự hỗ trợ của trình duyệt,một bảng chọn ngày có được hiển thị trong trường đầu vào.

<form>  
  Birthday:  
  <input type="date" name="bday">  
</form>

Bạn cũng có thể sử dụng thuộc tính min và max để thêm các hạn chế về ngày.

Ví dụ:

<form>  
  Enter a date before 1980-01-01:  
  <input type="date" name="bday" max="1979-12-31"><br>  
  Enter a date after 2000-01-01:  
  <input type="date" name="bday" min="2000-01-02"><br>  
</form>

* Input Type Datetime-local

<input type="datetime-local">quy định một trường đầu vào dạng ngày và giờ, không có múi giờ.

Tùy thuộc vào sự hỗ trợ của trình duyệt, một bảng chọn ngày có thể hiển thị trong trường đầu vào.

<form>  
  Birthday (date and time):  
  <input type="datetime-local" name="bdaytime">  
</form>

* Input Type Email

<input type="email">được sử dụng cho trường đầu vào mà chứa một địa chỉ email.

Tùy thuộc vào sự hỗ trợ của trình duyệt, đại chỉ email có thể tự động xác nhận khi gửi.

Một số điện thoại thông minh có thể nhận ra kiểu email, và thêm “.com” vào bàn phím để phù hợp với email đầu vào.

<form>  
  E-mail:  
  <input type="email" name="email">  
</form>

* Input Type File

<input type="file">định nghĩa một trường tập tin được chọn và một nút “Browse” để tải tập tin lên.

<form>  
  Select a file: <input type="file" name="myFile">  
</form>

* Input Type Month

<input type="month">cho phép người dùng lựa chọn một tháng và năm.

Tùy thuộc vào sự hỗ trợ của trình duyệt, một bảng chọn ngày có thể hiển thị ở trường đầu vào.

<form>  
  Birthday (month and year):  
  <input type="month" name="bdaymonth">  
</form>

* Input Type Number

<input type="number"> định nghĩa trường đầu vào là số.

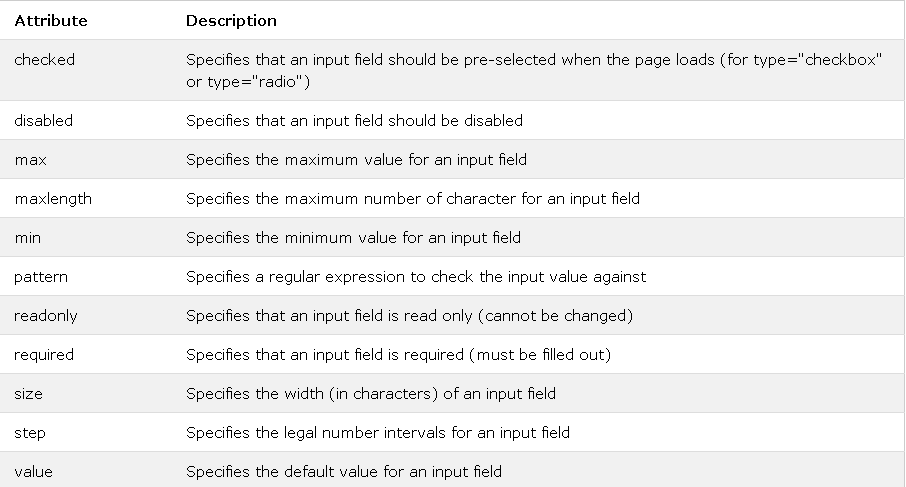
Bạn cũng có thể thiết lập hạn chế với những số được chấp thuận.

Ví dụ dưới đây hiển thị một trường đầu vào là số, nơi bạn có thể nhập giá trị từ 1-5:

<form>  
  Quantity (between 1 and 5):  
  <input type="number" name="quantity" min="1" max="5">  
</form>

* Input Restrictions

Dưới đây là danh sách một số hạn chế đầu vào phổ biến:



Ví dụ dưới đây hiển thị một trường số đầu vào, nơi bạn có thể nhập giá trị từ 0-100, bước nhảy là 10. Giá trị định là 30 :

<form>  
  Quantity:  
  <input type="number" name="points" min="0" max="100" step="10" value="30">  
</form>

* Input Type Range

<input type="range"> định nghĩa một điều khiển để nhập một số có giá trị chính xác là không quan trọng (như một điều khiển thanh trượt ). Phạm vi mặc định là từ 0 đến 100. Tuy nhiên, bạn có thể thiết lập các hạn chế về những con số với thuộc tính min, max và step.

Ví dụ :

<form>  
  <input type="range" name="points" min="0" max="10">  
</form>

* Input Type Search

<input type="search"> được dùng cho các trường tìm kiếm ( một trường tìm kiếm hoạt động như một trường văn bản thường).

<form>  
  Search Google:  
  <input type="search" name="googlesearch">  
</form>

* Input Type Tel

<input type="tel">được sử dụng cho trường đầu vào chứa một số điện thoại.

<form>  
  Telephone:  
  <input type="tel" name="phone" pattern="[0-9]{3}-[0-9]{2}-[0-9]{3}">  
</form>

* Input Type Time

<input type="time"> cho phép người dùng chọn một thời gian (không có múi giờ).

Tùy thuộc vào sự hỗ trợ của trình duyệt, một bảng chọn thời gian có thể hiện thị trong trường đầu vào.

Ví dụ :

<form>  
  Select a time:  
  <input type="time" name="usr\_time">  
</form>

* Input Type Url

 <input type="url">được sử dụng cho trường đầu vào chứa một địa chỉ url.

Tùy thuộc vào sự hỗ trợ của trình duyệt, trường url có thể tự động xác nhận khi gửi.

Một số điện thoại thông minh có thể nhận dạng url, và thêm “.com” vào bàn phím để phù hợp với đầu vào url.

<form>  
  Add your homepage:  
  <input type="url" name="homepage">  
</form>

* Input Type Week

<input type="week"> cho phép người dùng chọn một tuần và một năm.

Tùy thuộc vào sự hỗ trợ của trình duyệt, một bảng chọn ngày có thể xuất hiện trong trường đầu vào.

<form>  
  Select a week:  
  <input type="week" name="week\_year">  
</form>

* Bài tập HTML: Trong form dưới đây, hãy thêm một trường đầu vào cho kiểu văn bản, với tên là “username”.

<form action="/action\_page.php">  
<>  
</form>

#### **HTML Input Attributes**

* Thuộc tính value

Thuộc tính value xác định giá trị ban đầu cho một trường đầu vào :

<form action="">  
  First name:<br>  
  <input type="text" name="firstname" value="John">  
</form>

* Thuộc tính readonly

Thuộc tính readonly xác định rằng trường đầu vào là chỉ đọc (không thể thay đổi) :

<form action="">  
  First name:<br>  
  <input type="text" name="firstname" value="John" readonly>  
</form>

* Thuộc tính disabled

Thuộc tính disable xác định rằng trường đầu vào bị vô hiệu hóa.

Một trường đầu vào disable không thể sử dụng và nhấp vào được, và giá trị của nó sẽ không được gửi đi khi submit form :

<form action="">  
  First name:<br>  
  <input type="text" name="firstname" value="John" disabled>  
</form>

* Thuộc tính size

Thuộc tính size xác định kích thước cho trường đầu vào :

<form action="">  
  First name:<br>  
  <input type="text" name="firstname" value="John" size="40">  
</form>

* Thuộc tính maxlength

Thuộc tính maxlength xác định chiều dài tối đa cho phép của trường đàu vào :

<form action="">  
  First name:<br>  
  <input type="text" name="firstname" maxlength="10">  
</form>

Với một thuộc tính maxlength, trường đầu vào sẽ không chấp nhận nhiều hơn số ký tự được cho phép.

Các thuộc tính maxlength không cung cấp bất kỳ thông tin phản hồi. Nếu bạn muốn thông báo với người dùng, bạn phải viết mã Javascript.

* Thuộc tính HTML5

HTML5 đã thêm một số thuộc tính cho thẻ <input> sau :

* Autocomplete
* Autofocus
* Form
* Formaction
* Formenctype
* Formmethod
* Formnovalidate
* Formtarget
* Height and width
* List
* Min and max
* Multiple
* Pattern (regexp)
* Placeholder
* Required
* Step

Và một số thuộc tính cho thẻ <form> :

* Autocomplete
* Novalidate

Thuộc tính autocomplete

Thuộc tính autocomplete xác định liệu một form hay một trường đầu vào nên có autocomple bật hay tắt.

Khi autocomplete bật, trình duyệt tự động hoàn thành các giá trị đầu vào dựa trên các giá trị mà người dùng đã nhập.

Có thể có autocomplete “on” đối với form, và “off” cho các trường đầu vào cụ thể, hoặc ngược lại.

Thuộc tính autocomplete làm việc với <form> và kiểu <input> : text search, url, tel, email, password, datepicker, range, và color.

Ví dụ một form HTML với autocomplete”on” ( và “off” cho một trường đầu vào) :

<form action="/action\_page.php" autocomplete="on">  
  First name:<input type="text" name="fname"><br>  
  Last name: <input type="text" name="lname"><br>  
  E-mail: <input type="email" name="email" autocomplete="off"><br>  
  <input type="submit">  
</form>

Trong một số trình duyệt, có thể bạn cần kích hoạt chức năng autocomplete để oàn thành việc này.

Thuộc tính novalidate

Thuộc tính novalidate là một thuộc tính của <form>.

Khi sử dụng, novalidate xác định rằng các dữ liệu form không nrrn được xác nhận khi submit.

Ví dụ : chỉ ra form không được xác nhận khi submit:

<form action="/action\_page.php" novalidate>  
  E-mail: <input type="email" name="user\_email">  
  <input type="submit">  
</form>

Thuộc tính autofocus

Thuộc tính autofocus xác định rằng trường đầu vào sẽ tự động nổi bật khi tải trang.

Ví dụ : trường đầu vào “Firstname” tự động nổi bật khi tải trang:

First name:<input type="text" name="fname" autofocus>

Thuộc tính form

Thuộc tính form xác định một hoặc nhiều mẫu gửi mà các phần tử <input> thuộc về.

<form action="/action\_page.php" id="form1">  
  First name: <input type="text" name="fname"><br>  
  <input type="submit" value="Submit">  
</form>  
  
Last name: <input type="text" name="lname" form="form1">

Thuộc tính formaction

Thuộc tính formaction xác định URL của tập tin đó sẽ xử lý việc kiểm soát đầu vào khi biểu mẫu được gửi.

Thuộc tính formaction ghi đè lên thuộc tính action của phần tử <form>.

Thuộc tính formaction thường được dùng với type= “submit” và type= “image”.

<form action="/action\_page.php">  
  First name: <input type="text" name="fname"><br>  
  Last name: <input type="text" name="lname"><br>  
  <input type="submit" value="Submit"><br>  
  <input type="submit" formaction="/action\_page2.php"  
  value="Submit as admin">  
</form>

Thuộc tính formenctype

Thuộc tính formenctype quy định cụ thể cách dữ liệu form nên được mã hóa khi gửi (chỉ dùng cho form với mothod= “post”).

Thuộc tính formenctype ghi đè thuộc tính enctype của phần tử <form>.

Thuộc tính formenctype được sử dụng với type =”submit” và type= “image”.

<form action="/action\_page\_binary.asp" method="post">  
  First name: <input type="text" name="fname"><br>  
  <input type="submit" value="Submit">  
  <input type="submit" formenctype="multipart/form-data"  
  value="Submit as Multipart/form-data">  
</form>

Thuộc tính formmethod

Thuộc tính formmethod định nghĩa phương thức HTTP để gửi form-data đến URL thực thi.

Thuộc tính formmethod ghi đè thuộc tính method của phần tử <form>.

Thuộc tính formmethod có thể sử dụng với type= “submit” và type= “image”.

<form action="/action\_page.php" method="get">  
  First name: <input type="text" name="fname"><br>  
  Last name: <input type="text" name="lname"><br>  
  <input type="submit" value="Submit">  
  <input type="submit" formmethod="post" value="Submit using POST">  
</form>

Thuộc tính formnovalidate

Thuộc tính formnovalidate ghi đè thuộc tính novalidate của phần tử <form>.

Thuộc tính formnovalidate có thể sử dụng với type= “submit”.

<form action="/action\_page.php">  
  E-mail: <input type="email" name="userid"><br>  
  <input type="submit" value="Submit"><br>  
  <input type="submit" formnovalidate value="Submit without validation">  
</form>

Thuộc tính formtarget

Thuộc tính formtarget chỉ định một tên hay một từ khóa cho biết nơi để hiển thị câu trả lời mà nhận được sau khi nộp mẫu đơn,

Thuộc tính formtarget ghi đè thuộc tính target của phần tử <form>.

Thuộc tính formtarget có thể sử dụng với type= “submit” và type= “image”.

<form action="/action\_page.php">  
  First name: <input type="text" name="fname"><br>  
  Last name: <input type="text" name="lname"><br>  
  <input type="submit" value="Submit as normal">  
  <input type="submit" formtarget="\_blank"  
  value="Submit to a new window">  
</form>

Thuộc tính height và width

Thuộc tính height và width xác định chiều cao và chiều rộng của một phần tử <input type= “image”>.

Luôn xác định kích thước của hình ảnh. Nếu trình duyệt không biết kích thước, trang sẽ nhấp nháy trong khi tải hình ảnh.

<input type="image" src="img\_submit.gif" alt="Submit" width="48" height="48">

Thuộc tính list

Thuộc tính list chỉ đến một phần tử <datalist> chứa các lựa chọn được xác định trước cho một phần tử <input>.

<input list="browsers">  
<datalist id="browsers">  
  <option value="Internet Explorer">  
  <option value="Firefox">  
  <option value="Chrome">  
  <option value="Opera">  
  <option value="Safari">  
</datalist>

Thuộc tính min và max

Thuộc tính min và max xác định giá trị tối thiểu và tối đa cho một phần tử <input>.

Thuộc tính min và max làm việc với các loại đầu vào : number, range, date, datetime-local, month, time and week.

Enter a date before 1980-01-01:  
<input type="date" name="bday" max="1979-12-31">  
  
Enter a date after 2000-01-01:  
<input type="date" name="bday" min="2000-01-02">  
  
Quantity (between 1 and 5):  
<input type="number" name="quantity" min="1" max="5">

Thuộc tính multiple

Thuộc tính multiple xác định rằng người dùng được phép nhập nhiều hơn một giá trị trong phần tử <input>.

Thuộc tính multiple làm việc với các loại đầu vào : email, và file.

Select images: <input type="file" name="img" multiple>

Thuộc tính pattern

Thuộc tính pattern chỉ định một biểu thức chính quy mà các giá trị của phần tử <input> được kiểm tra đối chiếu.

Thuộc tính pattern làm việc với các loại đầu vào : text, search, url, tel, email, và password.

Sử dụng thuộc tính title toàn cục để mô tả mô hình để giúp người sử dụng.

Country code: <input type="text" name="country\_code" pattern="[A-Za-z]{3}" title="Three letter country code">

Thuộc tính placeholder

Thuộc tính placeholder chỉ định một gợi ý mô tả các giá trị kỳ vọng của một trường đầu vào (một giá trị mẫu hoặc một mô tả ngắn về định dạng).

Các gợi ý được hiển thị trong trường đầu vào trước khi người dùng nhập vào một giá trị.

Thuộc tính placeholder làm việc với các loại đầu vào : text, search, url, tel, email, và password.

<input type="text" name="fname" placeholder="First name">

Thuộc tính required

Thuộc tính required xác định rằng trường đầu vào phải được điền đầy đủ trước khi nộp mẫu đơn.

Thuộc tính required làm việc với các loại đầu vào : text, search, url, tel, email, password, date pickers, number, checkbox, radio, và file.

Username: <input type="text" name="usrname" required>

Thuộc tính step

Thuộc tính step xác định khoảng số pháp lý cho một phần tử <input>.

Ví dụ: nếu step= “3”, những số hợp lệ có thể là -3,0,3,6…

Thuộc tính step có thể được sử dụng cùng thuộc tính max và min để tạo một vùng các giá trị hợp lệ.

Thuộc tính step làm việc với các loại đầu vào :number, range, date, dayetime-local, month, time and week.

<input type="number" name="points" step="3">

* Bài tập HTML

Trong trường đầu vào dưới đây, thêm thuộc tính placeholder “Your name here”.

<form action="/action\_page.php">  
<input type="text" >  
</form>

### HTML Examples

<!DOCTYPE html>

<html>

<body>

<h2title="I'm a header">The title Attribute</h2>

<p title="I'm a tooltip">

Mouse over this paragraph, to display the title attribute as a tooltip.

</p>

</body>

</html>

### HTML References

Các thẻ HTML

|  |  |
| --- | --- |
| Thẻ | Mô tả |
| [<!--...-->](https://www.w3schools.com/tags/tag_comment.asp) | Định nghĩa một comment |
| [<!DOCTYPE>](https://www.w3schools.com/tags/tag_doctype.asp) | Định nghĩa loại tài liệu |
| [<a>](https://www.w3schools.com/tags/tag_a.asp) | Định nghĩa một siêu liên kết |
| [<abbr>](https://www.w3schools.com/tags/tag_abbr.asp) | Định nghĩa từ viết tắt hoặc tóm tắt một nội dung nào đó. |
| [<acronym>](https://www.w3schools.com/tags/tag_acronym.asp) | Không hỗ trợ trong HTML5. Thay thế sử dụng <abbr>.  Định nghĩa một từ viết tắt. |
| [<address>](https://www.w3schools.com/tags/tag_address.asp) | Định nghĩa thông tin liên lạc cho các tác giả/ chủ sở hữu của một tài liệu. |
| [<applet>](https://www.w3schools.com/tags/tag_applet.asp) | Không hỗ trợ trong HTML5. Thay thế sử dụng <embed> hoặc <object>.  Định nghĩa nhúng một applet. |
| [<area>](https://www.w3schools.com/tags/tag_area.asp) | Định nghĩa vùng bên trong của một ánh xạ hình ảnh. |
| [<article>](https://www.w3schools.com/tags/tag_article.asp) | Định nghĩa một bài báo |
| [<aside>](https://www.w3schools.com/tags/tag_aside.asp) | Định nghĩa nội dung nằm bên cạnh nội dung của trang |
| [<audio>](https://www.w3schools.com/tags/tag_audio.asp) | Định nghĩa một nội dung âm thanh. |
| [<b>](https://www.w3schools.com/tags/tag_b.asp) | Định nghĩa chữ in đậm. |
| [<base>](https://www.w3schools.com/tags/tag_base.asp) | Chỉ định URL cơ sở/target cho tất cả các URL tương đối trong một tài liệu |
| [<basefont>](https://www.w3schools.com/tags/tag_basefont.asp) | Không hỗ trợ trong HTML5. Sử dụng CSS để thay thế.  Chỉ định màu mặc định, kích thước và phông chữ cho tất cả phần văn bản trong tài liệu. |
| [<bdi>](https://www.w3schools.com/tags/tag_bdi.asp) | Tách biệt một phần của văn bản có thể được định dạng theo một hướng khác nhau từ văn bản khác bên ngoài nó |
| [<bdo>](https://www.w3schools.com/tags/tag_bdo.asp) | Ghi đè hướng văn bản hiện hành |
| [<big>](https://www.w3schools.com/tags/tag_big.asp) | Không hỗ trợ trong HTML5. Dùng CSS để thay thế.  Xác định văn bản lớn. |
| [<blockquote>](https://www.w3schools.com/tags/tag_blockquote.asp) | Định nghĩa một phần được trích dẫn từ nguồn khác |
| [<body>](https://www.w3schools.com/tags/tag_body.asp) | Định nghĩa than của tài liệu |
| [<br>](https://www.w3schools.com/tags/tag_br.asp) | Định nghĩa một ngắt dòng đơn |
| [<button>](https://www.w3schools.com/tags/tag_button.asp) | Định nghĩa một nút bấm |
| [<canvas>](https://www.w3schools.com/tags/tag_canvas.asp) | Được sử dụng để vẽ đồ họa, thông qua một ngôn ngữ script (thường là JavaScript) |
| [<caption>](https://www.w3schools.com/tags/tag_caption.asp) | Định nghĩa một chú thích trong bảng |
| [<center>](https://www.w3schools.com/tags/tag_center.asp) | Không hỗ trợ trong HTML5. Sử dụng CSS để thay thế.  Xác định văn bản làm trung tâm. |
| [<cite>](https://www.w3schools.com/tags/tag_cite.asp) | Định nghĩa nội dung được trích dẫn như một quyển sách, bài báo, một video,… |
| [<code>](https://www.w3schools.com/tags/tag_code.asp) | Định nghĩa một đoạn mã máy tính |
| [<col>](https://www.w3schools.com/tags/tag_col.asp) | Chỉ định thuộc tính cột cho mỗi cột trong một <colgroup> |
| [<colgroup>](https://www.w3schools.com/tags/tag_colgroup.asp) | Chỉ định một nhóm gồm một hoặc nhiều cột trong một bảng để định dạng |
| [<datalist>](https://www.w3schools.com/tags/tag_datalist.asp) | Định nghĩa một danh sách các giá trị tùy chọn được định nghĩa trước cho phần tử <input> |
| [<dd>](https://www.w3schools.com/tags/tag_dd.asp) | Được dùng để mô tả một từ khóa/ giá trị trong một danh sách mô tả |
| [<del>](https://www.w3schools.com/tags/tag_del.asp) | Định nghĩa văn bản đã bị xóa từ một tài liệu |
| [<details>](https://www.w3schools.com/tags/tag_details.asp) | Xác định các chi tiết khác mà người dùng có thể xem hoặc ẩn. |
| [<dfn>](https://www.w3schools.com/tags/tag_dfn.asp) | Mô tả định nghĩa của một từ |
| [<dialog>](https://www.w3schools.com/tags/tag_dialog.asp) | Định nghĩa một hộp thoại hoặc một cửa sổ |
| [<dir>](https://www.w3schools.com/tags/tag_dir.asp) | Không hỗ trợ trong HTML5. Sử dụng <ul> để thay thế.  Định nghĩa một danh sách thư mục. |
| [<div>](https://www.w3schools.com/tags/tag_div.asp) | Định nghĩa một phần trong tài liệu. |
| [<dl>](https://www.w3schools.com/tags/tag_dl.asp) | Định nghĩa một danh sách mô tả |
| [<dt>](https://www.w3schools.com/tags/tag_dt.asp) | Định nghĩa một từ khóa/tên trong danh sách mô tả. |
| [<em>](https://www.w3schools.com/tags/tag_em.asp) | Định nghĩa nhấn mạnh văn bản |
| [<embed>](https://www.w3schools.com/tags/tag_embed.asp) | Định nghĩa một container chứa một ứng dụng mở rộng(không phải HTML) |
| [<fieldset>](https://www.w3schools.com/tags/tag_fieldset.asp) | Nhóm lên nhóm các phần tử liên quan trong form |
| [<figcaption>](https://www.w3schools.com/tags/tag_figcaption.asp) | Định nghĩa một chú thích cho thẻ <figure> |
| [<figure>](https://www.w3schools.com/tags/tag_figure.asp) | Chỉ định nội dung độc lập |
| [<font>](https://www.w3schools.com/tags/tag_font.asp) | Không hỗ trợ trong HTML5. Sử dụng CSS để thay thế.  Xác định phông chữ, màu sắc , kích thước cho văn bản |
| [<footer>](https://www.w3schools.com/tags/tag_footer.asp) | Định nghĩa phần footer cho một tài liệu hoặc một đoạn |
| [<form>](https://www.w3schools.com/tags/tag_form.asp) | Định nghĩa một form HTML cho người dùng nhập vào. |
| [<frame>](https://www.w3schools.com/tags/tag_frame.asp) | Không hỗ trợ trong HTML5.  Định nghĩa một cửa sổ (khung) trong một khung |
| [<frameset>](https://www.w3schools.com/tags/tag_frameset.asp) | Không hỗ trợ trong HTML5.  Định nghĩa một tập hợp các frame |
| [<h1> to <h6>](https://www.w3schools.com/tags/tag_hn.asp) | Định nghĩa tiêu đề HTML |
| [<head>](https://www.w3schools.com/tags/tag_head.asp) | Xác định các thông tin về tài liệu |
| [<header>](https://www.w3schools.com/tags/tag_header.asp) | Định nghĩa một tiêu đề cho một tài liệu hoặc một phần |
| [<hr>](https://www.w3schools.com/tags/tag_hr.asp) | Sử dụng để phân tách các phần nội dung trong HTML |
| [<html>](https://www.w3schools.com/tags/tag_html.asp) | Định nghĩa gốc của một tài liệu HTML |
| [<i>](https://www.w3schools.com/tags/tag_i.asp) | Định nghĩa một phần chữ nghiêng cho văn bản |
| [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp) | Định nghĩa một khung nội tuyến. |
| [<img>](https://www.w3schools.com/tags/tag_img.asp) | Định nghĩa một hình ảnh |
| [<input>](https://www.w3schools.com/tags/tag_input.asp) | Định nghĩa một control nhập dữ liệu |
| [<ins>](https://www.w3schools.com/tags/tag_ins.asp) | Định nghĩa một đoạn văn bản đã được chèn thêm thay thế vào tài liệu |
| [<kbd>](https://www.w3schools.com/tags/tag_kbd.asp) | Là một thẻ dùng để nhấn mạnh, dùng để định nghĩa đầu vào bàn phím |
| [<label>](https://www.w3schools.com/tags/tag_label.asp) | Định nghĩa một nhãn cho một phần tử <input> |
| [<legend>](https://www.w3schools.com/tags/tag_legend.asp) | Định nghĩa một chú thích cho một phần tử <fieldset> |
| [<li>](https://www.w3schools.com/tags/tag_li.asp) | Định nghĩa một danh sách |
| [<link>](https://www.w3schools.com/tags/tag_link.asp) | Xác định mối quan hệ giữa một tài liệu và một nguồn lực bên ngoài (thường được sử dụng để liên kết đến style sheet) |
| [<main>](https://www.w3schools.com/tags/tag_main.asp) | Chỉ định các nội dung chính của một tài liệu |
| [<map>](https://www.w3schools.com/tags/tag_map.asp) | Định nghĩa một ánh xạ hình ảnh ở phía client |
| [<mark>](https://www.w3schools.com/tags/tag_mark.asp) | Định nghĩa đánh dấu/làm nổi bật văn bản |
| [<meta>](https://www.w3schools.com/tags/tag_meta.asp) | Định nghĩa siêu dữ liệu về một tài liệu HTML |
| [<meter>](https://www.w3schools.com/tags/tag_meter.asp) | Định nghĩa một thước đo lường với một khoảng đã biết |
| [<nav>](https://www.w3schools.com/tags/tag_nav.asp) | Định nghĩa liên kết điều hướng |
| [<noframes>](https://www.w3schools.com/tags/tag_noframes.asp) | Không hỗ trợ trong HTML5.  Định nghĩa một nội dung thay thế cho người dùng nếu trình duyệt không hỗ trợ khung nhìn. |
| [<noscript>](https://www.w3schools.com/tags/tag_noscript.asp) | Định nghĩa một nội dung thay thế cho người dùng mà trình duyệt phía client không hỗ trợ script |
| [<object>](https://www.w3schools.com/tags/tag_object.asp) | Định nghĩa một đối tượng nhúng vào trang |
| [<ol>](https://www.w3schools.com/tags/tag_ol.asp) | Định nghĩa một danh sách có sắp xếp |
| [<optgroup>](https://www.w3schools.com/tags/tag_optgroup.asp) | Định nghĩa một nhóm các tùy chọn liên quan trong một danh sách drop-down |
| [<option>](https://www.w3schools.com/tags/tag_option.asp) | Định nghĩa một lựa chọn trong danh sách drop-down |
| [<output>](https://www.w3schools.com/tags/tag_output.asp) | Xác định kết quả của một phép tính |
| [<p>](https://www.w3schools.com/tags/tag_p.asp) | Định nghĩa một đoạn văn bản |
| [<param>](https://www.w3schools.com/tags/tag_param.asp) | Định nghĩa một tham số cho một đối tượng |
| [<picture>](https://www.w3schools.com/tags/tag_picture.asp) | Xác định một vùng chứa cho nhiều tài nguyên hình ảnh |
| [<pre>](https://www.w3schools.com/tags/tag_pre.asp) | Định nghĩa văn bản định dạng sẵn |
| [<progress>](https://www.w3schools.com/tags/tag_progress.asp) | Miêu tả quá trình xử lý công việc bằng một thanh progressbar, thường được kết hợp với code Javascript |
| [<q>](https://www.w3schools.com/tags/tag_q.asp) | Định nghĩa dấu nháy kép xung quanh phần văn bản |
| [<rp>](https://www.w3schools.com/tags/tag_rp.asp) | Định nghĩa những gì thể hiện trong các trình duyệt không hỗ trợ các chú thích |
| [<rt>](https://www.w3schools.com/tags/tag_rt.asp) | Định nghĩa diễn giải/phát âm của các ký tự (đối với kiểu chữ Đông Á) |
| [<ruby>](https://www.w3schools.com/tags/tag_ruby.asp) | Định nghĩa một chú thích (đối với kiểu chữ Đông Á) |
| [<s>](https://www.w3schools.com/tags/tag_s.asp) | Định nghĩa văn bản đó không còn đúng |
| [<samp>](https://www.w3schools.com/tags/tag_samp.asp) | Hiển thị kết quả đầu ra của một chương trình máy tính |
| [<script>](https://www.w3schools.com/tags/tag_script.asp) | Định nghĩa một kịch bản phía máy khách |
| [<section>](https://www.w3schools.com/tags/tag_section.asp) | Định nghĩa một phần trong một tài liệu |
| [<select>](https://www.w3schools.com/tags/tag_select.asp) | Định nghĩa một danh sách drop-down |
| [<small>](https://www.w3schools.com/tags/tag_small.asp) | Định nghĩa văn bản chữ nhỏ |
| [<source>](https://www.w3schools.com/tags/tag_source.asp) | Định nghĩa các nguồn đa phương tiện cho các phần tử đa phương tiện như <video> và <audio> |
| [<span>](https://www.w3schools.com/tags/tag_span.asp) | Định nghĩa một phần trong một tài liệu |
| [<strike>](https://www.w3schools.com/tags/tag_strike.asp) | Không hỗ trợ trong HTML5. Sử dụng <del> hoặc <s> thay thế.  Định nghĩa chữ gạch ngang |
| [<strong>](https://www.w3schools.com/tags/tag_strong.asp) | Định nghĩa văn bản in đậm, ý nghĩa nhấn mạnh độ quan trọng |
| [<style>](https://www.w3schools.com/tags/tag_style.asp) | Định nghĩa kiểu cho một tài liệu |
| [<sub>](https://www.w3schools.com/tags/tag_sub.asp) | Định nghĩa văn bản giống như chỉ số dưới |
| [<summary>](https://www.w3schools.com/tags/tag_summary.asp) | Hiển thị tiêu đề cho phần tử <detail> |
| [<sup>](https://www.w3schools.com/tags/tag_sup.asp) | Hiển thị văn bản giống số mũ |
| [<svg>](https://www.w3schools.com/tags/tag_svg.asp) | Xác định một vùng chứa cho đồ họa SVG |
| [<table>](https://www.w3schools.com/tags/tag_table.asp) | Định nghĩa một bảng |
| [<tbody>](https://www.w3schools.com/tags/tag_tbody.asp) | Nhóm các nội dung trong một bảng |
| [<td>](https://www.w3schools.com/tags/tag_td.asp) | Định nghĩa một ô trong một bảng |
| [<template>](https://www.w3schools.com/tags/tag_template.asp) | Xác định một mẫu |
| [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Xác định một ô nhập dữ liệu văn bản có nhiều dòng |
| [<tfoot>](https://www.w3schools.com/tags/tag_tfoot.asp) | Nhóm các nội dung footer trong một bảng |
| [<th>](https://www.w3schools.com/tags/tag_th.asp) | Định nghĩa một ô tiêu đề trong một bảng |
| [<thead>](https://www.w3schools.com/tags/tag_thead.asp) | Nhóm các nội dung tiêu đề trong một bảng |
| [<time>](https://www.w3schools.com/tags/tag_time.asp) | Định nghĩa ngày/giờ |
| [<title>](https://www.w3schools.com/tags/tag_title.asp) | Định nghĩa một tiêu đề cho tài liệu |
| [<tr>](https://www.w3schools.com/tags/tag_tr.asp) | Định nghĩa một hàng trong một bảng |
| [<track>](https://www.w3schools.com/tags/tag_track.asp) | Định nghĩa nội dung mô tả như chú thích, bình luận, hoặc các loại văn bản khác cho tập tin đa phương tiện (<video> và <audio>) |
| [<tt>](https://www.w3schools.com/tags/tag_tt.asp) | Không hỗ trợ trong HTML5. Sử dụng CSS để thay thế.  Xác định văn bản của máy |
| [<u>](https://www.w3schools.com/tags/tag_u.asp) | Sử dụng để tạo chữ gạch chân cho phần văn bản |
| [<ul>](https://www.w3schools.com/tags/tag_ul.asp) | Định nghĩa một danh sách k có thứ tự |
| [<var>](https://www.w3schools.com/tags/tag_var.asp) | Định nghĩa một biến |
| [<video>](https://www.w3schools.com/tags/tag_video.asp) | Định nghĩa một video hoặc movie |
| [<wbr>](https://www.w3schools.com/tags/tag_wbr.asp) | Định nghĩa một dấu ngắt dòng |

Các thuộc tính HTML :

|  |  |  |
| --- | --- | --- |
| Thuộc tính | Cho thẻ | Mô tả |
| [accept](https://www.w3schools.com/tags/att_accept.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp) | Chỉ định các loại tập tin mà máy chủ chấp nhận (chỉ dành cho thẻ input có type= “file”) |
| [accept-charset](https://www.w3schools.com/tags/att_accept-charset.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp) | Chỉ định kiểu mã ký tự được sử dụng cho form gửi dữ liệu đi. |
| [accesskey](https://www.w3schools.com/tags/att_accesskey.asp) | [Tất](https://www.w3schools.com/tags/ref_standardattributes.asp) cả các thẻ | Chỉ định một phím tắt để kích hoạt/focus vào một phần tử |
| [action](https://www.w3schools.com/tags/att_action.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp) | Chỉ định nơi để gửi dữ liệu trên form khi một form được kích nút gửi. |
| align | Không hỗ trợ trong HTML5 | Chỉ định việc căn chỉnh dữ liệu của các phần tử. thường sử dụng trong CSS. |
| [alt](https://www.w3schools.com/tags/att_alt.asp) | [<area>](https://www.w3schools.com/tags/tag_area.asp), [<img>](https://www.w3schools.com/tags/tag_img.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp) | Chỉ định văn bản thay thế cho các phần tử không hiển thị, ví dụ như phần tử img ta dùng alt để mô tả thông tin cho ảnh để các công cụ tìm kiếm dễ tìm thấy. |
| [async](https://www.w3schools.com/tags/att_async.asp) | [<script>](https://www.w3schools.com/tags/tag_script.asp) | Chỉ định kịch bản không đồng bộ (chỉ cho các script bên ngoài). |
| [autocomplete](https://www.w3schools.com/tags/att_autocomplete.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp) | Chỉ định một form hoặc một phần tử input có thiết lập thuộc tính autocomplete hay không, ví dụ như gõ vào ô textbox, nếu thiết lập autocomplete nó sẽ gợi ý các từ đã nhập trước đây. |
| [autofocus](https://www.w3schools.com/tags/att_autofocus.asp) | [<button>](https://www.w3schools.com/tags/tag_button.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<select>](https://www.w3schools.com/tags/tag_select.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Xác định các phần tử sẽ tự động nhận focus khi tải lại trang |
| [autoplay](https://www.w3schools.com/tags/att_autoplay.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Chỉ định âm thanh/video sẽ bắt đầu chạy khi nó đã sẵn sang. |
| bgcolor | Không hỗ trợ trong HTML5. | Xác định màu nền của một phần tử. Thường sử dụng trong CSS. |
| border | Không hỗ trợ trong HTML5. | Xác định độ rộng đường giới của một phần tử. Thường dùng trong CSS. |
| [charset](https://www.w3schools.com/tags/att_charset.asp) | [<meta>](https://www.w3schools.com/tags/tag_meta.asp), [<script>](https://www.w3schools.com/tags/tag_script.asp) | Xác định bộ mã ký tự cho văn bản HTML |
| [checked](https://www.w3schools.com/tags/att_checked.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp) | Chỉ định một phần tử <input> là được chọn khi tải lại trang (đối với type = “checkbox” hoặc type = “radio”). |
| [cite](https://www.w3schools.com/tags/att_cite.asp) | <blockquote>, [<del>](https://www.w3schools.com/tags/tag_del.asp), [<ins>](https://www.w3schools.com/tags/tag_ins.asp), <q> | Chỉ định một đường dẫn URL giải thích việc trích dẫn/xóa/chèn văn bản vào. |
| [class](https://www.w3schools.com/tags/att_class.asp) | Tất cả các thẻ | Chỉ định một hoặc nhiều tên class cho một phần tử (đề cập đến 1 class trong 1 stylesheet). |
| color | Không hỗ trợ trong HTML5 | Xác định màu chữ của một phần tử. Thường dùng trong CSS. |
| [cols](https://www.w3schools.com/tags/att_cols.asp) | [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Xác định chiều rộng cột cho phần tử text area. |
| [colspan](https://www.w3schools.com/tags/att_colspan.asp) | [<td>](https://www.w3schools.com/tags/tag_td.asp), [<th>](https://www.w3schools.com/tags/tag_th.asp) | Xác định số cột của các ô trong bảng sẽ được gộp lại. |
| [content](https://www.w3schools.com/tags/att_content.asp) | [<meta>](https://www.w3schools.com/tags/tag_meta.asp) | Cung cấp giá trị liên quan với http-equiv hoặc thuộc tính name. |
| [contenteditable](https://www.w3schools.com/tags/att_contenteditable.asp) | Tất cả các thẻ | Xác định nội dung của một phần tử có thể biên tập được hay không. |
| [controls](https://www.w3schools.com/tags/att_controls.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Xác định trình điều khiển âm thanh/video sẽ được hiển thị (ví dụ như một nút play/pause…). |
| [coords](https://www.w3schools.com/tags/att_coords.asp) | [<area>](https://www.w3schools.com/tags/tag_area.asp) | Xác định tọa độ của một vùng. |
| [data](https://www.w3schools.com/tags/att_data.asp) | [<object>](https://www.w3schools.com/tags/tag_object.asp) | Chỉ định URL của một tài nguyên được sử dụng bởi một đối tượng. |
| [data-\*](https://www.w3schools.com/tags/att_data-.asp) | Tất cả các thẻ | Được sử dụng để lưu trữ dữ liệu tùy biến riêng cho các trang hoặc ứng dụng. |
| [datetime](https://www.w3schools.com/tags/att_datetime.asp) | [<del>](https://www.w3schools.com/tags/tag_del.asp), [<ins>](https://www.w3schools.com/tags/tag_ins.asp), [<time>](https://www.w3schools.com/tags/tag_time.asp) | Xác định thời gian. |
| [default](https://www.w3schools.com/tags/att_default.asp) | [<track>](https://www.w3schools.com/tags/tag_track.asp) | Là thuộc tính kiểu Boolean, xác định track này là được kích hoạt nếu người dùng không chỉ rõ một track khác thích hợp hơn. |
| [defer](https://www.w3schools.com/tags/att_defer.asp) | [<script>](https://www.w3schools.com/tags/tag_script.asp) | Chỉ định rằng script đã được thực thi khi trang đã hoàn thành phân tích (chỉ dành cho kịch bản bên ngoài). |
| [dir](https://www.w3schools.com/tags/att_dir.asp) | Tất cả các thẻ | Chỉ định hướng của văn bản cho nội dung trong một phần tử ví dụ trái sang phải, phải qua trái… |
| [dirname](https://www.w3schools.com/tags/att_dirname.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Xác định hướng của văn bản khi được submit. |
| [disabled](https://www.w3schools.com/tags/att_disabled.asp) | [<button>](https://www.w3schools.com/tags/tag_button.asp), [<fieldset>](https://www.w3schools.com/tags/tag_fieldset.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), <optgroup>, [<option>](https://www.w3schools.com/tags/tag_option.asp), [<select>](https://www.w3schools.com/tags/tag_select.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Chỉ định một hoặc một nhóm các phần tử sẽ bị vô hiệu hóa. |
| [download](https://www.w3schools.com/tags/att_download.asp) | [<a>](https://www.w3schools.com/tags/tag_a.asp), [<area>](https://www.w3schools.com/tags/tag_area.asp) | Chỉ định đích được tải tới khi người dùng nhấn vào một liên kết. |
| [draggable](https://www.w3schools.com/tags/att_draggable.asp) | [Tất](https://www.w3schools.com/tags/ref_standardattributes.asp) cả các thẻ | Chỉ định một phần tử có thể kéo thả hay không. |
| [dropzone](https://www.w3schools.com/tags/att_dropzone.asp) | [Tất](https://www.w3schools.com/tags/ref_standardattributes.asp) cả các thẻ | Chỉ định dữ liệu được kéo là sao chép, di chuyển, hoặc liên kết khi thả ra (dùng cho sự kiện kéo thả). |
| [enctype](https://www.w3schools.com/tags/att_enctype.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp) | Specifies how the form-data should be encoded when submitting it to the server (only for method="post") |
| [for](https://www.w3schools.com/tags/att_for.asp) | <label>, [<output>](https://www.w3schools.com/tags/tag_output.asp) | Chỉ định các phần tử của form của một nhãn hoặc một phép tính có ràng buộc tới. |
| [form](https://www.w3schools.com/tags/att_form.asp) | [<button>](https://www.w3schools.com/tags/tag_button.asp), [<fieldset>](https://www.w3schools.com/tags/tag_fieldset.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), <label>, [<meter>](https://www.w3schools.com/tags/tag_meter.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<output>](https://www.w3schools.com/tags/tag_output.asp), [<select>](https://www.w3schools.com/tags/tag_select.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Chỉ định các phần tử được chứa trong một form có tên nào đó. |
| [formaction](https://www.w3schools.com/tags/att_formaction.asp) | [<button>](https://www.w3schools.com/tags/tag_button.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp) | Chỉ định nơi để gửi dữ liệu của form khi một form được submit. Chỉ dành cho type= “submit”. |
| [headers](https://www.w3schools.com/tags/att_headers.asp) | [<td>](https://www.w3schools.com/tags/tag_td.asp), [<th>](https://www.w3schools.com/tags/tag_th.asp) | Chỉ định một hoặc nhiều ô ở phần tiêu đề của bảng có liên quan đến với các ô dữ liệu trong bảng. |
| [height](https://www.w3schools.com/tags/att_height.asp) | <canvas>, [<embed>](https://www.w3schools.com/tags/tag_embed.asp), [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp), [<img>](https://www.w3schools.com/tags/tag_img.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Chỉ định chiều cao của phần tử. |
| [hidden](https://www.w3schools.com/tags/att_hidden.asp) | [Tất](https://www.w3schools.com/tags/ref_standardattributes.asp) cả các thẻ | Là thuộc tính Boolean, chỉ định một phần tử không được hiển thị. |
| [high](https://www.w3schools.com/tags/att_high.asp) | [<meter>](https://www.w3schools.com/tags/tag_meter.asp) | Xác định giá trị một khoảng nào đó và được xem như là một giá trị độ cao. |
| [href](https://www.w3schools.com/tags/att_href.asp) | [<a>](https://www.w3schools.com/tags/tag_a.asp), [<area>](https://www.w3schools.com/tags/tag_area.asp), [<base>](https://www.w3schools.com/tags/tag_base.asp), [<link>](https://www.w3schools.com/tags/tag_link.asp) | Chỉ định URL của trang liên kết đi tới. |
| [hreflang](https://www.w3schools.com/tags/att_hreflang.asp) | [<a>](https://www.w3schools.com/tags/tag_a.asp), [<area>](https://www.w3schools.com/tags/tag_area.asp), [<link>](https://www.w3schools.com/tags/tag_link.asp) | Chỉ định ngôn ngữ của tài liệu được liên kết. |
| [http-equiv](https://www.w3schools.com/tags/att_http-equiv.asp) | [<meta>](https://www.w3schools.com/tags/tag_meta.asp) | Cung cấp phần tiêu đề HTTP cho các thông tin/giá trị của thuộc tính nội dung. |
| [id](https://www.w3schools.com/tags/att_id.asp) | [Tất](https://www.w3schools.com/tags/ref_standardattributes.asp) cả các thẻ | Chỉ định một id duy nhất cho một phần tử. |
| [ismap](https://www.w3schools.com/tags/att_ismap.asp) | [<img>](https://www.w3schools.com/tags/tag_img.asp) | Chỉ định một hình ảnh như một server-side image-map. |
| [kind](https://www.w3schools.com/tags/att_kind.asp) | [<track>](https://www.w3schools.com/tags/tag_track.asp) | Chỉ định loại text cho phần tử track. |
| [label](https://www.w3schools.com/tags/att_label.asp) | [<track>](https://www.w3schools.com/tags/tag_track.asp), [<option>](https://www.w3schools.com/tags/tag_option.asp), [<optgroup>](https://www.w3schools.com/tags/tag_optgroup.asp) | Chỉ định tiêu đề của text cho phần tử track. |
| [lang](https://www.w3schools.com/tags/att_lang.asp) | [Tất](https://www.w3schools.com/tags/ref_standardattributes.asp) cả các thẻ | Chỉ định ngôn ngữ cho phần nội dung của phần tử. |
| [list](https://www.w3schools.com/tags/att_list.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp) | Đề cập đến phần tử <datalisr> mà chứa các giá trị tùy chọn định nghĩa trước cho một phần tử input. |
| [loop](https://www.w3schools.com/tags/att_loop.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Chỉ định các âm thanh/video sẽ lặp lại một lần nữa mỗi khi nó chạy hết. |
| [low](https://www.w3schools.com/tags/att_low.asp) | [<meter>](https://www.w3schools.com/tags/tag_meter.asp) | Xác định khoảng giá trị thấp cho phần tử meter |
| [max](https://www.w3schools.com/tags/att_max.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<meter>](https://www.w3schools.com/tags/tag_meter.asp), [<progress>](https://www.w3schools.com/tags/tag_progress.asp) | Xác định giá trị lớn nhất cho phần tử. |
| [maxlength](https://www.w3schools.com/tags/att_maxlength.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Xác định số lượng ký tự tối đa cho phép trong một phần tử. |
| [media](https://www.w3schools.com/tags/att_media.asp) | [<a>](https://www.w3schools.com/tags/tag_a.asp), [<area>](https://www.w3schools.com/tags/tag_area.asp), [<link>](https://www.w3schools.com/tags/tag_link.asp), [<source>](https://www.w3schools.com/tags/tag_source.asp), [<style>](https://www.w3schools.com/tags/tag_style.asp) | Chỉ định những gì truyền thông/thiết bị các tài liệu liên quan được tối ưu hóa cho. |
| [method](https://www.w3schools.com/tags/att_method.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp) | Chỉ định các phương thức HTTP để sử dụng khi gửi form-data. |
| [min](https://www.w3schools.com/tags/att_min.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<meter>](https://www.w3schools.com/tags/tag_meter.asp) | Chỉ định một giá trị tối thiểu. |
| [multiple](https://www.w3schools.com/tags/att_multiple.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<select>](https://www.w3schools.com/tags/tag_select.asp) | Chỉ định rằng người dùng có thể nhập vào nhiều hơn một giá trị. |
| [muted](https://www.w3schools.com/tags/att_muted.asp) | [<video>](https://www.w3schools.com/tags/tag_video.asp), [<audio>](https://www.w3schools.com/tags/tag_audio.asp) | Chỉ định rằng đầu ra âm thanh của video nên được tắt. |
| [name](https://www.w3schools.com/tags/att_name.asp) | [<button>](https://www.w3schools.com/tags/tag_button.asp), [<fieldset>](https://www.w3schools.com/tags/tag_fieldset.asp), [<form>](https://www.w3schools.com/tags/tag_form.asp), [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<map>](https://www.w3schools.com/tags/tag_map.asp), [<meta>](https://www.w3schools.com/tags/tag_meta.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<output>](https://www.w3schools.com/tags/tag_output.asp), [<param>](https://www.w3schools.com/tags/tag_param.asp), [<select>](https://www.w3schools.com/tags/tag_select.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Chỉ định tên của phần tử. |
| [novalidate](https://www.w3schools.com/tags/att_novalidate.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp) | Chỉ định form không cần kiểm tra tính hợp lệ (validate) trước khi được submit. |
| [onabort](https://www.w3schools.com/tags/att_onabort.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<embed>](https://www.w3schools.com/tags/tag_embed.asp), [<img>](https://www.w3schools.com/tags/tag_img.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script này được chạy khi hủy bỏ. |
| [onafterprint](https://www.w3schools.com/tags/att_onafterprint.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script này được chạy sau khi các tài liệu được in. |
| [onbeforeprint](https://www.w3schools.com/tags/att_onbeforeprint.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script này được chạy trước khi các tài liệu được in. |
| [onbeforeunload](https://www.w3schools.com/tags/att_onbeforeunload.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script này được chạy khi tài liệu sắp được gỡ bỏ. |
| [onblur](https://www.w3schools.com/tags/att_onblur.asp) | Có ở tất cả các phần tử | Script này được chạy khi phần tử mất focus. |
| [oncanplay](https://www.w3schools.com/tags/att_oncanplay.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<embed>](https://www.w3schools.com/tags/tag_embed.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script này được chạy khi một tập tin đã sẵn sang để bắt đầu (file đã nạp đủ trong bộ để chạy) |
| [oncanplaythrough](https://www.w3schools.com/tags/att_oncanplaythrough.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script này được chạy khi một tập tin có thể được chạy theo tất cả các cách khác nhau đến kết thúc mà không dừng lại nạp vào bộ đệm. |
| [onchange](https://www.w3schools.com/tags/att_onchange.asp) | Có ở tất cả các phần tử | Script này được chạy khi giá trị của phần tử bị thay đổi. |
| [onclick](https://www.w3schools.com/tags/att_onclick.asp) | Có ở tất cả các phần tử. | Script này được chạy khi giá trị của phần tử được kích vào. |
| [oncontextmenu](https://www.w3schools.com/tags/att_oncontextmenu.asp) | Có ở tất cả các phần tử. | Script này được chạy khi một trình đơn ngữ cảnh được kích hoạt. |
| [oncopy](https://www.w3schools.com/tags/att_oncopy.asp) | Có ở tất cả các phần tử. | Script này được chạy khi nội dung của phần tử đang được sao chép. |
| [oncuechange](https://www.w3schools.com/tags/att_oncuechange.asp) | [<track>](https://www.w3schools.com/tags/tag_track.asp) | Script này được chạy khi có dấu hiệu thay đổi trong phần tử <track> |
| [oncut](https://www.w3schools.com/tags/att_oncut.asp) | Có ở tất cả các phần tử. | Script này được chạy khi nội dung của phần tử đang được cắt giảm. |
| [ondblclick](https://www.w3schools.com/tags/att_ondblclick.asp) | Có ở tất cả các phần tử. | Script này được chạy khi phần tử này được nhấn đúp chuột. |
| [ondrag](https://www.w3schools.com/tags/att_ondrag.asp) | Có ở tất cả các phần tử. | Script này được chạy ở phần cuối của một hoạt động kéo. |
| [ondragend](https://www.w3schools.com/tags/att_ondragend.asp) | Có ở tất cả các phần tử. | Script này được chạy khi kết thúc một hoạt động kéo. |
| [ondragenter](https://www.w3schools.com/tags/att_ondragenter.asp) | Có ở tất cả các phần tử. | Script được chạy khi một phần tử đã được kéo đến một mục tiêu thả hợp lệ. |
| [ondragleave](https://www.w3schools.com/tags/att_ondragleave.asp) | Có ở tất cả các phần tử. | Script này được chạy khi một phần tử rời khỏi một mục tiêu thả hợp lệ. |
| [ondragover](https://www.w3schools.com/tags/att_ondragover.asp) | Có ở tất cả các phần tử. | Script này được chạy khi một phần tử đang được kéo qua một mục tiêu thả hợp lệ. |
| [ondragstart](https://www.w3schools.com/tags/att_ondragstart.asp) | Có ở tất cả các phần tử. | Script này được chạy khi bắt đầu một hoạt động kéo. |
| [ondrop](https://www.w3schools.com/tags/att_ondrop.asp) | Có ở tất cả các phần tử. | Script được chạy khi yếu tố kéo đang được giảm. |
| [ondurationchange](https://www.w3schools.com/tags/att_ondurationchange.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script này được chạy khi thay đổi chiều dài của những phương tiện truyền thông. |
| [onemptied](https://www.w3schools.com/tags/att_onemptied.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script này được chạy khi một cái gì xấu xảy ra và tập tin là đột nhiên không có sẵn (như bất ngờ ngắt kết nối). |
| [onended](https://www.w3schools.com/tags/att_onended.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi các phương tiện truyền thông đã được kết thúc ( một sự kiện hữu ích cho những thông điệp như “cảm ơn đã nghe”). |
| [onerror](https://www.w3schools.com/tags/att_onerror.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<body>](https://www.w3schools.com/tags/tag_body.asp), [<embed>](https://www.w3schools.com/tags/tag_embed.asp), [<img>](https://www.w3schools.com/tags/tag_img.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<script>](https://www.w3schools.com/tags/tag_script.asp), [<style>](https://www.w3schools.com/tags/tag_style.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi một lỗi xảy ra. |
| [onfocus](https://www.w3schools.com/tags/att_onfocus.asp) | Có ở tất cả các phần tử. | Script được chạy khi các yếu tố được tập trung. |
| [onhashchange](https://www.w3schools.com/tags/att_onhashchange.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi đã có những thay đổi trong phần neo của một URL. |
| [oninput](https://www.w3schools.com/tags/att_oninput.asp) | Có ở tất cả các phần tử. | Script được chạy khi các yếu tố đầu vào được sử dụng. |
| [oninvalid](https://www.w3schools.com/tags/att_oninvalid.asp) | Có ở tất cả các phần tử. | Script được chạy khi các phần tử không hợp lệ. |
| [onkeydown](https://www.w3schools.com/tags/att_onkeydown.asp) | Có ở tất cả các phần tử. | Script được chạy khi người dùng nhấn một phím bất kỳ. |
| [onkeypress](https://www.w3schools.com/tags/att_onkeypress.asp) | Có ở tất cả các phần tử. | Script được chạy khi người dùng nhấn một phím bất kỳ. |
| [onkeyup](https://www.w3schools.com/tags/att_onkeyup.asp) | Có ở tất cả các phần tử. | Script được chạy khi người dùng nhả một phím. |
| [onload](https://www.w3schools.com/tags/att_onload.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp), [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp), [<img>](https://www.w3schools.com/tags/tag_img.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<link>](https://www.w3schools.com/tags/tag_link.asp), [<script>](https://www.w3schools.com/tags/tag_script.asp), [<style>](https://www.w3schools.com/tags/tag_style.asp) | Script được chạy khi các phần tử được tải xong. |
| [onloadeddata](https://www.w3schools.com/tags/att_onloadeddata.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy ki dữ liệu truyền thông được tải. |
| [onloadedmetadata](https://www.w3schools.com/tags/att_onloadedmetadata.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi siêu dữ liệu (như kích thước và thời gian) được nạp. |
| [onloadstart](https://www.w3schools.com/tags/att_onloadstart.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy như các tập tin để bắt đầu tải trước khi bất cứ điều gì là thực sự nạp. |
| [onmousedown](https://www.w3schools.com/tags/att_onmousedown.asp) | Có ở tất cả các phần tử. | Script được chạy khi một nút chuột được nhấn xuống trên một phần tử. |
| [onmousemove](https://www.w3schools.com/tags/att_onmousemove.asp) | Có ở tất cả các phần tử. | Script được chạy khi con trỏ chuột di chuyển trên một phần tử. |
| [onmouseout](https://www.w3schools.com/tags/att_onmouseout.asp) | Có ở tất cả các phần tử. | Script được chạy khi một con trỏ chuột di chuyển khỏi các phần tử. |
| [onmouseover](https://www.w3schools.com/tags/att_onmouseover.asp) | Có ở tất cả các phần tử. | Script được chạy khi một con trỏ chuột di chuyển trên một phần tử. |
| [onmouseup](https://www.w3schools.com/tags/att_onmouseup.asp) | Có ở tất cả các phần tử. | Script được chạy khi một nút chuột được nhấn trên một phần tử. |
| [onmousewheel](https://www.w3schools.com/tags/att_onmousewheel.asp) | Có ở tất cả các phần tử. | Script được chạy khi một bánh xe chuột đang được cuộn trên một phần tử. |
| [onoffline](https://www.w3schools.com/tags/att_onoffline.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi trình duyệt bắt đầu làm việc chế độ offline |
| [ononline](https://www.w3schools.com/tags/att_ononline.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi trình duyệt bắt đầu làm việc chế độ online. |
| onpagehide | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi người dùng điều hướng ra khỏi một trang. |
| [onpageshow](https://www.w3schools.com/tags/att_onpageshow.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi người dùng điều hướng đến một trang. |
| [onpaste](https://www.w3schools.com/tags/att_onpaste.asp) | Có ở tất cả các phần tử. | Script được chạy khi người dùng dán một số nội dung vào một phần tử. |
| [onpause](https://www.w3schools.com/tags/att_onpause.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi tập tin đa phương tiện bị tạm dừng hoặc bởi người sử dụng hoặc lập trình. |
| [onplay](https://www.w3schools.com/tags/att_onplay.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi tập tin đa phương tiện đã sẵn sàng để chạy. |
| [onplaying](https://www.w3schools.com/tags/att_onplaying.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi tập tin đa phương tiện thực sự đã bắt đầu chạy. |
| onpopstate | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi lịch sử của cửa sổ thay đổi. |
| [onprogress](https://www.w3schools.com/tags/att_onprogress.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi trình duyệt đang trong quá trình xử lý nhận dữ liệu đa phương tiện. |
| [onratechange](https://www.w3schools.com/tags/att_onratechange.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy mỗi khi thay đổi tốc độ phát lại (giống như khi người dùng chuyển sang một chế độ chuyển động chậm hoặc chế độ chuyển tiếp nhanh). |
| [onreset](https://www.w3schools.com/tags/att_onreset.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp) | Script được chạy khi nút reset trên form được kích chuột. |
| [onresize](https://www.w3schools.com/tags/att_onresize.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi cửa sổ trình duyệt bị thay đổi kích thước. |
| [onscroll](https://www.w3schools.com/tags/att_onscroll.asp) | Có ở tất cả các phần tử. | Script được chạy khi thanh cuộn của phần tử được cuộn. |
| [onsearch](https://www.w3schools.com/tags/att_onsearch.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp) | Script được chạy khi người dùng viết một cái gì đó trong trường search (<input="search">) |
| [onseeked](https://www.w3schools.com/tags/att_onseeked.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi các thuộc tính seeking được thiết lập là false để chỉ tìm kiếm đã kết thúc. |
| [onseeking](https://www.w3schools.com/tags/att_onseeking.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi các thuộc tính seeking được thiết lập là true để chỉ tìm kiếm được kích hoạt. |
| [onselect](https://www.w3schools.com/tags/att_onselect.asp) | Có ở tất cả các phần tử. | Script được chạy khi các phần tử được chọn. |
| [onstalled](https://www.w3schools.com/tags/att_onstalled.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi trình duyệt là không thể lấy dữ liệu đa phương tiện vì bất kỳ lý do gì. |
| onstorage | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi vùng lưu trữ web bị cập nhật. |
| [onsubmit](https://www.w3schools.com/tags/att_onsubmit.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp) | Script được chạy khi form được submit. |
| [onsuspend](https://www.w3schools.com/tags/att_onsuspend.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi việc nạp dữ liệu đa phương tiện bị dừng lại trước khi nó hoàn thành việc tải về với bất kỳ lý do gì |
| [ontimeupdate](https://www.w3schools.com/tags/att_ontimeupdate.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi các vị trí đang chạy bị thay đổi (như khi người dùng chuyển nhanh đến một điểm khác nhau của tệp tin đa phương tiện) |
| [ontoggle](https://www.w3schools.com/tags/att_ontoggle.asp) | [<details>](https://www.w3schools.com/tags/tag_details.asp) | Script được chạy khi người dùng mở hoặc đóng phần tử <detail>. |
| [onunload](https://www.w3schools.com/tags/att_onunload.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi một trang đã được giải phóng (hoặc cửa sổ trình duyệt đã bị đóng). |
| [onvolumechange](https://www.w3schools.com/tags/att_onvolumechange.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy mỗi khi âm lượng được thay đổi. |
| [onwaiting](https://www.w3schools.com/tags/att_onwaiting.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi tập tin đa phương tiện đang tạm dừng nhưng dự kiến sẽ tiếp tục (giống như tập tin đa phương tiện dừng để đợi bộ đệm nhiều dữ liệu hơn). |
| [onwheel](https://www.w3schools.com/tags/att_onwheel.asp) | Có ở tất cả các phần tử. | Script được chạy khi con lăn ở giữa chuột cuộn lên hoặc xuống trên một phần tử. |
| [open](https://www.w3schools.com/tags/att_open.asp) | [<details>](https://www.w3schools.com/tags/tag_details.asp) | Chỉ định phần thông tin chi tiết sẽ được mở (hiển thị) cho người dùng. |
| [optimum](https://www.w3schools.com/tags/att_optimum.asp) | [<meter>](https://www.w3schools.com/tags/tag_meter.asp) | Chỉ định giá trị tối ưu cho phần tử meter. |
| [pattern](https://www.w3schools.com/tags/att_pattern.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp) | Chỉ định một biểu thức chính quy mà giá trị của phần tử <input> được kiểm tra. |
| [placeholder](https://www.w3schools.com/tags/att_placeholder.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Chỉ định một gợi ý ngắn mô tả chức năng cho phần tử. |
| [poster](https://www.w3schools.com/tags/att_poster.asp) | [<video>](https://www.w3schools.com/tags/tag_video.asp) | Chỉ định một hình ảnh được hiển thị trong khi video được tải về, hoặc cho đến khi người dùng chạm vào nút play. |
| [preload](https://www.w3schools.com/tags/att_preload.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Chỉ định các file âm thanh/video có được tải khi trang được tải hay không. |
| [readonly](https://www.w3schools.com/tags/att_readonly.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Chỉ định rằng các phần tử chỉ được đọc. |
| [rel](https://www.w3schools.com/tags/att_rel.asp) | [<a>](https://www.w3schools.com/tags/tag_a.asp), [<area>](https://www.w3schools.com/tags/tag_area.asp), [<link>](https://www.w3schools.com/tags/tag_link.asp) | Xác định mối quan hệ giữa các tài liệu hiện hành và các tài liệu được liên kết. |
| [required](https://www.w3schools.com/tags/att_required.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<select>](https://www.w3schools.com/tags/tag_select.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Chỉ định rằng phần tử này phải được điền đầy đủ trước khi form được submit. |
| [reversed](https://www.w3schools.com/tags/att_reversed.asp) | [<ol>](https://www.w3schools.com/tags/tag_ol.asp) | Chỉ định rằng thứ tự danh sách nên giảm dần (9,8,7…). |
| [rows](https://www.w3schools.com/tags/att_rows.asp) | [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Xác định số lượng dòng có trong một TextArea. |
| [rowspan](https://www.w3schools.com/tags/att_rowspan.asp) | [<td>](https://www.w3schools.com/tags/tag_td.asp), [<th>](https://www.w3schools.com/tags/tag_th.asp) | Xác định số lượng ô trong một hàng của bảng được gộp. |
| [sandbox](https://www.w3schools.com/tags/att_sandbox.asp) | [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp) | Cho phép thêm các hạn chế đối với các nội dung trong một <iframe> |
| [scope](https://www.w3schools.com/tags/att_scope.asp) | [<th>](https://www.w3schools.com/tags/tag_th.asp) | Chỉ định tiêu đề của một ô là một tiêu đề cho một cột, một hàng, hay một nhóm của các cột hoặc các hàng. |
| [selected](https://www.w3schools.com/tags/att_selected.asp) | [<option>](https://www.w3schools.com/tags/tag_option.asp) | Chỉ định một giá trị được chọn mặc định khi tải trang. |
| [shape](https://www.w3schools.com/tags/att_shape.asp) | [<area>](https://www.w3schools.com/tags/tag_area.asp) | Chỉ định hình dạng của phần tử area. |
| [size](https://www.w3schools.com/tags/att_size.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<select>](https://www.w3schools.com/tags/tag_select.asp) | Xác định chiều rộng hay số lượng ký tự (cho thẻ <input>) hoặc số lượng hiển thị các giá trị tùy chọn (cho thẻ <select>). |
| [sizes](https://www.w3schools.com/tags/att_sizes.asp) | [<img>](https://www.w3schools.com/tags/tag_img.asp), [<link>](https://www.w3schools.com/tags/tag_link.asp), [<source>](https://www.w3schools.com/tags/tag_source.asp) | Chỉ định kích thước của tài nguyên được liên kết tới. |
| [span](https://www.w3schools.com/tags/att_span.asp) | [<col>](https://www.w3schools.com/tags/tag_col.asp), [<colgroup>](https://www.w3schools.com/tags/tag_colgroup.asp) | Xác định số lượng cột được gộp lại thành một. |
| [spellcheck](https://www.w3schools.com/tags/att_spellcheck.asp) | Tất cả các thẻ | Chỉ định các phần tử có kiểm tra lỗi chính tả và ngữ pháp hay không. |
| [src](https://www.w3schools.com/tags/att_src.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<embed>](https://www.w3schools.com/tags/tag_embed.asp), [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp), [<img>](https://www.w3schools.com/tags/tag_img.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<script>](https://www.w3schools.com/tags/tag_script.asp), [<source>](https://www.w3schools.com/tags/tag_source.asp), [<track>](https://www.w3schools.com/tags/tag_track.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Chỉ định URL của tập tin đa phương tiện. |
| [srcdoc](https://www.w3schools.com/tags/att_srcdoc.asp) | [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp) | Chỉ định các nội dung HTML của trang hiển thị trong thẻ <iframe>. |
| [srclang](https://www.w3schools.com/tags/att_srclang.asp) | [<track>](https://www.w3schools.com/tags/tag_track.asp) | Chỉ định ngôn ngữ dữ liệu văn bản của track (được yêu cầu type = “subtitle”). |
| [srcset](https://www.w3schools.com/tags/att_source_srcset.asp) | [<img>](https://www.w3schools.com/tags/tag_img.asp), [<source>](https://www.w3schools.com/tags/tag_source.asp) | Chỉ định URL của hình ảnh để sử dụng trong các tình huống khác nhau. |
| [start](https://www.w3schools.com/tags/att_start.asp) | [<ol>](https://www.w3schools.com/tags/tag_ol.asp) | Chỉ định giá trị bắt đầu của danh sách đầu mục là số. |
| [step](https://www.w3schools.com/tags/att_step.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp) | Chỉ định bước nhảy với dữ liệu số cho trường input. |
| [style](https://www.w3schools.com/tags/att_style.asp) | Tất cả các thẻ | Chỉ định style CSS inline ho một phần tử |
| [tabindex](https://www.w3schools.com/tags/att_tabindex.asp) | Tất cả các thẻ | Chỉ định thứ tự khi ấn phím tab cho một phần tử. |
| [target](https://www.w3schools.com/tags/att_target.asp) | [<a>](https://www.w3schools.com/tags/tag_a.asp), [<area>](https://www.w3schools.com/tags/tag_area.asp), [<base>](https://www.w3schools.com/tags/tag_base.asp), [<form>](https://www.w3schools.com/tags/tag_form.asp) | Xác định đích khi mở các tài liệu, liên quan hoặc nơi các form được submit. |
| [title](https://www.w3schools.com/tags/att_title.asp) | Tất cả các thẻ | Thêm thông tin về một phần tử. |
| [translate](https://www.w3schools.com/tags/att_translate.asp) | Tất cả các thẻ | Chỉ định nội dung của một phần tử phải được dịch hay không. |
| [type](https://www.w3schools.com/tags/att_type.asp) | [<a>](https://www.w3schools.com/tags/tag_a.asp), [<button>](https://www.w3schools.com/tags/tag_button.asp), [<embed>](https://www.w3schools.com/tags/tag_embed.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<link>](https://www.w3schools.com/tags/tag_link.asp), [<menu>](https://www.w3schools.com/tags/tag_menu.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<script>](https://www.w3schools.com/tags/tag_script.asp), [<source>](https://www.w3schools.com/tags/tag_source.asp), [<style>](https://www.w3schools.com/tags/tag_style.asp) | Chỉ định kiểu của phần tử. |
| [usemap](https://www.w3schools.com/tags/att_usemap.asp) | [<img>](https://www.w3schools.com/tags/tag_img.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp) | Chỉ định việc ánh xạ một hình ảnh ở phía client-side image-map. |
| [value](https://www.w3schools.com/tags/att_value.asp) | [<button>](https://www.w3schools.com/tags/tag_button.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<li>](https://www.w3schools.com/tags/tag_li.asp), [<option>](https://www.w3schools.com/tags/tag_option.asp), [<meter>](https://www.w3schools.com/tags/tag_meter.asp), [<progress>](https://www.w3schools.com/tags/tag_progress.asp), [<param>](https://www.w3schools.com/tags/tag_param.asp) | Xác định giá trị của các phần tử. |
| [width](https://www.w3schools.com/tags/att_width.asp) | <canvas>, [<embed>](https://www.w3schools.com/tags/tag_embed.asp), [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp), [<img>](https://www.w3schools.com/tags/tag_img.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Xác định chiều rộng của phần tử. |
| [wrap](https://www.w3schools.com/tags/att_wrap.asp) | [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Chỉ định cách hiển thị văn bản trong textarea. |

### HTML5

#### HTML5 Intro

* Có gì mới trong HTML5?

Khai báo DOCTYPE cho HTML5 rất đơn giản:

<!DOCTYPE html>

Khai báo mã hóa ký tự (bộ ký tự) cũng rất đơn giản:

<meta charset="UTF-8">

Ví dụ HTML5:

<!DOCTYPE html>  
<html>  
<head>  
<meta charset="UTF-8">  
<title>Title of the document</title>  
</head>  
<body>  
Content of the document......  
</body>  
</html>

Mã hóa ký tự mặc định trong HTML5 là UTF-8



* Các yếu tố mới trong html mới

Các yếu tố HTML5 mới thú vị nhất là:

New yếu tố ngữ nghĩa như <header>, <footer>, <article>, <section>.

Các thuộc tính mới của các thành phần biểu mẫu như số, ngày, thời gian, lịch và phạm vi.

Yếu tố đồ họa mới : <svg>và <canvas>.

Các yếu tố đa phương tiện mới : <audio>và <video>.

Trong chương tiếp theo, Hỗ trợ HTML5 , bạn sẽ tìm hiểu cách "dạy" các trình duyệt cũ hơn để xử lý các phần tử HTML "không xác định" (mới).

* API HTML5 mới (Giao diện lập trình ứng dụng)

Các API mới thú vị nhất trong HTML5 là:

- Định vị địa lý HTML

- Kéo và thả HTML

- Lưu trữ cục bộ HTML

- Bộ nhớ cache ứng dụng HTML

- Công nhân web HTML

- HTML SSE

* Các phần tử bị xóa trong HTML5

Các phần tử HTML4 sau đây đã bị xóa trong HTML5:



Trong chương Di chuyển HTML5 , bạn sẽ tìm hiểu cách dễ dàng di chuyển từ HTML4 sang HTML5.

* Lịch sử HTML

Kể từ những ngày đầu của World Wide Web, đã có nhiều phiên bản HTML:



#### HTML5 Support

* Hỗ trợ trình duyệt HTML5

HTML5 được hỗ trợ trong tất cả các trình duyệt hiện đại.

Ngoài ra, tất cả các trình duyệt, cũ và mới, tự động xử lý các phần tử không được nhận dạng dưới dạng phần tử nội tuyến.

Do đó, bạn có thể "dạy" các trình duyệt cũ hơn để xử lý các phần tử HTML "không xác định".

* Xác định các phần tử ngữ nghĩa là các phần tử khối

HTML5 định nghĩa tám yếu tố ngữ nghĩa mới . Tất cả đều là các yếu tố cấp khối .

Để bảo đảm hành vi chính xác trong các trình duyệt cũ hơn, bạn có thể đặt thuộc tính hiển thị CSS cho các thành phần HTML này thành chặn :

header, section, footer, aside, nav, main, article, figure {  
  display: block;  
}

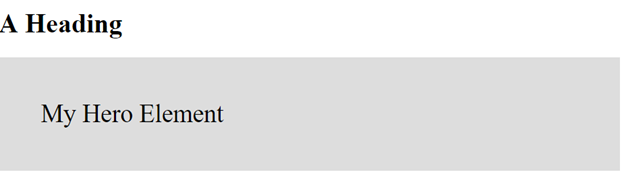
* Thêm các yếu tố mới vào HTML

Bạn cũng có thể thêm các yếu tố mới vào trang HTML bằng thủ thuật trình duyệt.

Ví dụ này thêm một yếu tố mới được gọi <myHero>vào trang HTML và xác định kiểu cho nó:

<!DOCTYPE html>  
<html>  
<head>  
<script>document.createElement("myHero")</script>  
<style>  
myHero {  
  display: block;  
  background-color: #dddddd;  
  padding: 50px;  
  font-size: 30px;  
}  
</style>  
</head>  
<body>  
<h1>A Heading</h1>  
<myHero>My Hero Element</myHero>  
</body>

</html>



Câu lệnh JavaScript document.createElement("myHero")("myHero")là cần thiết để tạo một phần tử mới trong IE 9 và trước đó.

* Sự cố với Internet Explorer 8
* Cú pháp cho HTML5Shiv

HTML5Shiv được đặt trong <head>thẻ.

HTML5Shiv là một tệp javascript được tham chiếu trong <script>thẻ.

Bạn nên sử dụng HTML5Shiv khi bạn đang sử dụng các yếu tố HTML5 mới như: <article>, <section>, <aside>, <nav>, <footer>

Bạn có thể tải xuống phiên bản HTML5shiv mới nhất từ ​​github hoặc tham khảo phiên bản CDN tại https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js

Cú pháp

<head>  
  <!--[if lt IE 9]>  
    <script src="/js/html5shiv.js"></script>  
  <![endif]-->  
</head>

* Ví dụ HTML5Shiv

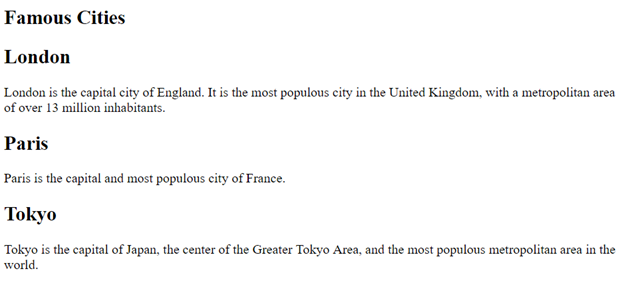
Nếu bạn không muốn tải xuống và lưu trữ HTML5Shiv trên trang web của mình, bạn có thể tham khảo phiên bản được tìm thấy trên trang web CDN.

Tập lệnh HTML5Shiv phải được đặt trong <head>phần tử, sau bất kỳ biểu định kiểu nào:

Ví dụ:

<!DOCTYPE html>  
<html>  
<head>  
<meta charset="UTF-8">  
<!--[if lt IE 9]>  
  <script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>  
<![endif]-->  
</head>  
<body>  
<section>  
<h1>Famous Cities</h1>  
<article>  
<h2>London</h2>  
<p>London is the capital city of England. It is the most populous city in the United Kingdom, with a metropolitan area of over 13 million inhabitants.</p>  
</article>  
<article>  
<h2>Paris</h2>  
<p>Paris is the capital and most populous city of France.</p>  
</article>  
<article>  
<h2>Tokyo</h2>  
<p>Tokyo is the capital of Japan, the center of the Greater Tokyo Area, and the most populous metropolitan area in the world.</p>  
</article>  
</section>  
</body>  
</html>

HIỂN THỊ



#### HTML5 New Elements

* Các yếu tố mới trong HTML5
* Các yếu tố cấu trúc / ngữ nghĩa mới

HTML5 cung cấp các yếu tố mới cho cấu trúc tài liệu tốt hơn

* Yếu tố hình thức mới

|  |  |
| --- | --- |
| Tag | Description |
| <datalist> | Specifies a list of pre-defined options for input controls |
| <output> | Defines the result of a calculation |

* Kiểu đầu vào mới.

|  |  |
| --- | --- |
| Kiểu đầu vào mới | Thuộc tính đầu vào mới |
| màu sắc  ngày  ngày giờ  datetime-local  e-mail  tháng  con số  phạm vi  Tìm kiếm  điện thoại  thời gian  url  tuần | tự động hoàn thành  tự động lấy nét  hình thức  hình thành  formenctype  hình thức  formnovalidate  nhắm mục tiêu  chiều cao và chiều rộng  danh sách  tối thiểu và tối đa  nhiều  mô hình (regrec)  giữ chỗ  cần thiết  bươc |

* HTML5 cú pháp và thuộc tính mới.

HTML5 cho phép bốn cú pháp khác nhau cho các thuộc tính.

|  |  |
| --- | --- |
| Type | Example |
| Empty | <input type="text" value="John" **disabled**> |
| Unquoted | <input type="text" **value=John**> |

Ví dụ này cho thấy các cú pháp khác nhau được sử dụng trong một

<input>

|  |  |
| --- | --- |
| Double-quoted | <input type="text" **value="John Doe"**> |
| Single-quoted | <input type="text" **value='John Doe'**> |

Trong HTML5, tất cả bốn cú pháp có thể được sử dụng, tùy thuộc vào những gì cần thiết cho thuộc tính.

* Đồ họa HTML5

|  |  |
| --- | --- |
| Tag | Description |
| <canvas> | Draw graphics, on the fly, via scripting (usually JavaScript) |
| <svg> | Draw scalable vector graphics |

* Yếu tố truyền thông mới

|  |  |
| --- | --- |
| Tag | Description |
| <audio> | Defines sound content |
| <embed> | Defines a container for an external (non-HTML) application |
| <source> | Defines multiple media resources for media elements (<video> and <audio>) |
| <track> | Defines text tracks for media elements (<video> and <audio>) |
| <video> | Defines video or movie |

#### HTML5 Semantics

* Các yếu tố ngữ nghĩa là gì?

Một yếu tố ngữ nghĩa mô tả rõ ràng ý nghĩa của nó đối với cả trình duyệt và nhà phát triển.

Ví dụ về các yếu tố phi ngữ nghĩa : <div>, <span>Không nói gì về nội dung của nó.

Ví dụ về ngữ nghĩa yếu tố: <form>, <table>, <article>định nghĩa rõ ràng nội dung của nó.

* Hỗ trợ trình duyệt

Các yếu tố ngữ nghĩa HTML5 được hỗ trợ trong tất cả các trình duyệt hiện đại: chrome, coccoc, safari, v.vv

* Các yếu tố ngữ nghĩa mới trong HTML5

Nhiều trang web chứa mã HTML như: <div id = "nav"><div class = "header"><div id = "footer">

để biểu thị điều hướng, tiêu đề và chân trang.

|  |
| --- |
| HTML5 cung cấp các yếu tố ngữ nghĩa mới để xác định các phần khác nhau của trang web: |

<bài viết>

<sang một bên>

<chi tiết>

<figcaption>

<hình>

<chân trang>

<tiêu đề>

<chính>

<đánh dấu>

<điều hướng>

<phần>

<tóm tắt>

<thời gian>

* Phần tử <phần> HTML5

Phần <section>tử xác định một phần trong tài liệu.

Theo tài liệu HTML5 của W3C: "Một phần là một nhóm nội dung theo chủ đề, thường có tiêu đề."

Một trang chủ thường có thể được chia thành các phần để giới thiệu, nội dung và thông tin liên lạc.

Ví dụ:

<section>  
  <h1>WWF</h1>  
  <p>The World Wide Fund for Nature (WWF) is....</p>  
</section>>

* Phần tử <bài viết> HTML5

Phần <article>tử chỉ định nội dung độc lập, khép kín.

Một bài viết nên có ý nghĩa riêng của nó, và nó có thể đọc nó độc lập với phần còn lại của trang web.

Ví dụ về nơi một <article>yếu tố có thể được sử dụng

- Diễn đàn bài

- Bài viết trên blog

- Bài viết trên báo

Ví dụ:

<article>  
  <h1>What Does WWF Do?</h1>  
  <p>WWF's mission is to stop the degradation of our planet's natural environment,  
  and build a future in which humans live in harmony with nature.</p>  
</article>

* Làm tổ <bài viết> trong <phần> hoặc Phó Versa?

Phần <article>tử chỉ định nội dung độc lập, khép kín.

Phần <section>tử xác định phần trong tài liệu.

Vì vậy, trên Internet, bạn sẽ tìm thấy các trang HTML có <section>các phần tử chứa <article>các phần tử và <article>các phần tử có chứa <section>các phần tử.

Bạn cũng sẽ tìm thấy các trang có <section>các phần tử chứa <section>các phần tử và <article>các phần tử có chứa article>các phần tử.

Ví dụ cho một tờ báo: Các môn thể thao <article>trong các môn thể thao phần , có thể có một kỹ thuật phần trong mỗi <article>

* Phần tử <tiêu đề> HTML5

Phần <header>tử chỉ định một tiêu đề cho một tài liệu hoặc phần.

Các <header>yếu tố nên được sử dụng như một thùng chứa cho nội dung giới thiệu.

Bạn có thể có một số <header>yếu tố trong một tài liệu.

Ví dụ sau định nghĩa một tiêu đề cho một bài viết:

<article>  
  <header>  
    <h1>What Does WWF Do?</h1>  
    <p>WWF's mission:</p>  
  </header>  
  <p>WWF's mission is to stop the degradation of our planet's natural environment,  
  and build a future in which humans live in harmony with nature.</p>  
</article>

* Phần tử <footer> HTML5

Phần <footer>tử chỉ định chân trang cho tài liệu hoặc phần.

Một <footer>phần tử nên chứa thông tin về phần tử chứa nó.

Một chân trang thường chứa tác giả của tài liệu, thông tin bản quyền, liên kết đến các điều khoản sử dụng, thông tin liên hệ, v.v.

Bạn có thể có một số <footer>yếu tố trong một tài liệu.

Ví dụ:

<footer>  
  <p>Posted by: Hege Refsnes</p>  
  <p>Contact information: <a href="mailto:someone@example.com">  
  someone@example.com</a>.</p>  
</footer>

HIỂN THỊ:



* Phần tử <nav> HTML5

Phần <nav>tử xác định một tập hợp các liên kết điều hướng.

Lưu ý rằng KHÔNG phải tất cả các liên kết của tài liệu phải nằm trong một <nav>phần tử. Phần <nav>tử chỉ dành cho khối chính của các liên kết điều hướng.

Ví dụ:

<nav>  
  <a href="/html/">HTML</a> |  
  <a href="/css/">CSS</a> |  
  <a href="/js/">JavaScript</a> |  
  <a href="/jquery/">jQuery</a>  
</nav>

HIỂN THỊ



* Phần tử <sid>>>

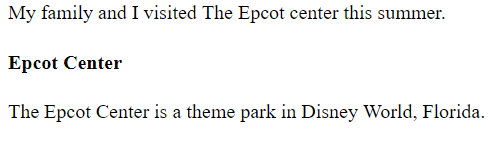
Phần <aside>tử xác định một số nội dung ngoài nội dung mà nó được đặt (như một thanh bên).

Các <aside>nội dung liên quan đến nội dung xung quanh.

Ví dụ:

<p>My family and I visited The Epcot center this summer.</p>  
<aside>  
  <h4>Epcot Center</h4>  
  <p>The Epcot Center is a theme park in Disney World, Florida.</p>  
</aside>

HIỂN THỊ:



* Các phần tử <hình> và <figcaption>

Mục đích của chú thích hình là để thêm một lời giải thích trực quan cho hình ảnh.

Trong HTML5, một hình ảnh và chú thích có thể được nhóm lại với nhau trong một thành <figure> phần:

Ví dụ:

<figure>  
  <img src="pic\_trulli.jpg" alt="Trulli">  
  <figcaption>Fig1. - Trulli, Puglia, Italy.</figcaption>  
</figure>

HIỂN THỊ:



* Tại sao các yếu tố ngữ nghĩa?

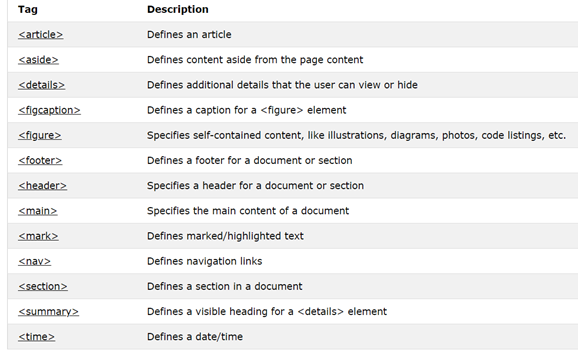
Với HTML4, các nhà phát triển đã sử dụng tên id / class của riêng họ để tạo thành phần kiểu: tiêu đề, đầu, cuối, chân trang, menu, điều hướng, chính, thùng chứa, nội dung, bài viết, thanh bên, topnav, v.v.

Điều này khiến các công cụ tìm kiếm không thể xác định đúng nội dung trang web.

Với các phần tử HTML5 mới ( <header><footer><nav><section><article>), điều này sẽ trở nên dễ dàng hơn.

Theo W3C, một Semantic Web: "Cho phép dữ liệu được chia sẻ và sử dụng lại trên các ứng dụng, doanh nghiệp và cộng đồng."

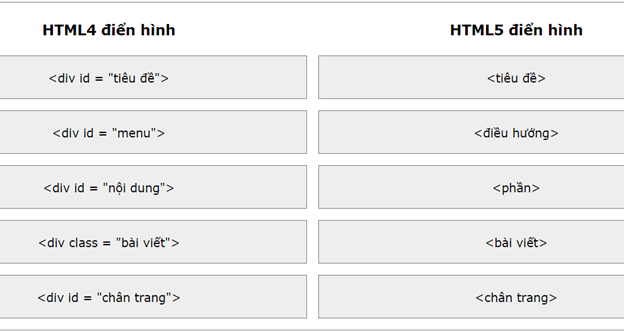
* Các yếu tố ngữ nghĩa trong HTML5



#### HTML5 Migration

* Di chuyển từ HTML4 sang HTML5

Chương này trình bày cách chuyển đổi trang HTML4 thành trang HTML5 mà không phá hủy bất kỳ nội dung hoặc cấu trúc gốc nào.



* Trang HTML4 điển hình

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">  
<html lang="en">  
<head>  
<meta http-equiv="Content-Type" content="text/html;charset=utf-8">  
<title>HTML4</title>  
<style>  
body {  
  font-family: Verdana,sans-serif;  
  font-size: 0.9em;  
}  
div#header, div#footer {  
  padding: 10px;  
  color: white;  
  background-color: black;  
}  
div#content {  
  margin: 5px;  
  padding: 10px;  
  background-color: lightgrey;  
}  
div.article {  
  margin: 5px;  
  padding: 10px;  
  background-color: white;  
}  
div#menu ul {  
  padding: 0;  
}  
div#menu ul li {  
  display: inline;  
  margin: 5px;  
}  
</style>  
</head>  
<body>  
<div id="header">  
  <h1>Monday Times</h1>  
</div>  
<div id="menu">  
  <ul>  
    <li>News</li>  
    <li>Sports</li>  
    <li>Weather</li>  
  </ul>  
</div>  
<div id="content">  
  <h2>News Section</h2>  
  <div class="article">  
    <h2>News Article</h2>  
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque in porta lorem. Morbi condimentum est nibh, et consectetur tortor feugiat at.</p>  
  </div>  
  <div class="article">  
    <h2>News Article</h2>  
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque in porta lorem. Morbi condimentum est nibh, et consectetur tortor feugiat at.</p>  
  </div>  
</div>  
<div id="footer">  
  <p>&amp;copy; 2016 Monday Times. All rights reserved.</p>  
</div>  
</body>  
</html>

* Thay đổi thành Tài liệu HTML5

Thay đổi từ:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

Sang tài liệu HTML5

<!DOCTYPE html>

* Thay đổi thành Mã hóa HTML5

Thay đổi thông tin mã hóa từ:

<meta http-equiv="Content-Type" content="text/html;charset=utf-8">

Sang mã hóa HTML5:

<meta charset="utf-8">

* Thêm HTML5Shiv

Ví dụ:

<!--[if lt IE 9]>  
  <script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>  
<![endif]-->

* Thay đổi thành các phần tử ngữ nghĩa HTML5

CSS hiện có chứa id và các lớp để tạo kiểu cho các phần tử:

Ví dụ:

body {  
  font-family: Verdana,sans-serif;  
  font-size: 0.9em;  
}  
div#header, div#footer {  
  padding: 10px;  
  color: white;  
  background-color: black;  
}  
div#content {  
  margin: 5px;  
  padding: 10px;  
  background-color: lightgrey;  
}  
div.article {  
  margin: 5px;  
  padding: 10px;  
  background-color: white;  
}  
div#menu ul {  
  padding: 0;  
}  
div#menu ul li {  
  display: inline;  
  margin: 5px;  
}

Thay thế bằng các kiểu CSS bằng nhau cho các yếu tố ngữ nghĩa HTML5:

body {  
  font-family: Verdana,sans-serif;  
  font-size: 0.9em;  
}  
header, footer {  
  padding: 10px;  
  color: white;  
  background-color: black;  
}  
section {  
  margin: 5px;  
  padding: 10px;  
  background-color: lightgrey;  
}  
article {  
  margin: 5px;  
  padding: 10px;  
  background-color: white;  
}  
nav ul {  
  padding: 0;  
}  
  
nav ul li {  
  display: inline;  
  margin: 5px;  
}

Cuối cùng, thay đổi các thành phần thành các yếu tố ngữ nghĩa HTML5:

<body>

<header>  
  <h1>Monday Times</h1>  
</header>  
<nav>  
  <ul>  
    <li>News</li>  
    <li>Sports</li>  
    <li>Weather</li>  
  </ul>  
</nav>  
<section>  
  <h2>News Section</h2>  
  <article>  
    <h2>News Article</h2>  
    <p>Lorem ipsum dolor sit amet..</p>  
  </article>  
  <article>  
    <h2>News Article</h2>  
    <p>Lorem ipsum dolor sit amet..</p>  
  </article>  
</section>  
<footer>  
  <p>&copy; 2014 Monday Times. All rights reserved.</p>  
</footer>  
</body>

* Sự khác biệt giữa <article><section>và <div>

Có một sự khác biệt (thiếu) khó hiểu trong tiêu chuẩn HTML5, giữa <article> <section> <div>.

Trong tiêu chuẩn HTML5, <section>phần tử được xác định là một khối các phần tử liên quan.

Phần <article> tử được định nghĩa là một khối hoàn chỉnh, khép kín của các phần tử liên quan.

Phần <div>tử được định nghĩa là một khối các phần tử con.

Làm thế nào để giải thích điều đó?

Trong ví dụ trên, chúng tôi đã sử dụng <section>như một container cho liên quan <article>.

Nhưng, chúng ta có thể đã sử dụng <article>như một container cho các bài viết.

#### HTML5 Style Guide

* Các quy ước mã hóa HTML

Các nhà phát triển web thường không chắc chắn về kiểu mã hóa và cú pháp sử dụng trong HTML.

Với XHTML, các nhà phát triển đã buộc phải viết mã hợp lệ và "được định dạng tốt".

HTML5 cẩu thả hơn một chút khi xác thực mã.

* Hãy thông minh và bằng chứng trong tương lai

Việc sử dụng nhất quán về phong cách giúp người khác dễ hiểu HTML của bạn hơn.

Trong tương lai, các chương trình như trình đọc XML có thể muốn đọc HTML của bạn.

Sử dụng cú pháp "gần với XHTML" được định dạng tốt có thể thông minh.

* Sử dụng đúng loại tài liệu

Luôn khai báo loại tài liệu là dòng đầu tiên trong tài liệu của bạn:

<!DOCTYPE html>

Nếu bạn muốn thống nhất với các thẻ chữ thường, bạn có thể sử dụng:

<!doctype html>

* Sử dụng tên phần tử chữ thường.

HTML5 cho phép trộn chữ hoa và chữ thường trong tên thành phần.

Nên sử dụng tên thành phần chữ thường vì:\

Trộn tên viết hoa và viết thường là xấu

* Các nhà phát triển thường sử dụng tên viết thường (như trong XHTML)
* Chữ thường trông sạch sẽ hơn
* Chữ thường dễ viết hơn
* Đóng tất cả các thành phần HTML
* Đóng các thành phần HTML trống

Trong HTML5, nó là tùy chọn để đóng các phần tử trống.

Tuy nhiên, dấu gạch chéo (/) được BẮT BUỘC trong XHTML và XML.

Nếu mong đợi phần mềm XML truy cập vào trang của mình, thì nên giữ dấu gạch chéo.

* Sử dụng tên thuộc tính chữ thường
* Trích dẫn giá trị thuộc tính

HTML5 cho phép các giá trị thuộc tính mà không có dấu ngoặc kép. Nên trích dẫn các giá trị thuộc tính vì:

* Các nhà phát triển thường trích dẫn các giá trị thuộc tính (như trong XHTML)
* Các giá trị được trích dẫn dễ đọc hơn
* Bạn PHẢI sử dụng dấu ngoặc kép nếu giá trị chứa khoảng trắng
* Thuộc tính hình ảnh

Luôn thêm altthuộc tính cho hình ảnh. Thuộc tính này rất quan trọng khi hình ảnh vì một số lý do không thể được hiển thị. Ngoài ra, luôn luôn xác định chiều rộng và chiều cao hình ảnh. Nó giảm nhấp nháy vì trình duyệt có thể dành không gian cho hình ảnh trước khi tải.

* Dấu cách và dấu bằng

HTML5 cho phép không gian xung quanh các dấu bằng. Nhưng không gian ít dễ đọc hơn và nhóm các thực thể tốt hơn với nhau.

* Tránh các dòng mã dài

Khi sử dụng trình soạn thảo HTML, thật bất tiện khi cuộn sang phải và trái để đọc mã HTML.

Cố gắng tránh các dòng mã dài hơn 80 ký tự.

* Dòng trống và thụt lề

Không thêm dòng trống mà không có lý do.

Để dễ đọc, hãy thêm các dòng trống để tách các khối mã lớn hoặc logic.

Để dễ đọc, thêm hai khoảng cách thụt. Không sử dụng phím tab.

Không sử dụng các dòng trống và thụt đầu dòng không cần thiết. Không cần thiết phải thụt lề mọi phần tử:

* Bỏ qua <html> và <body>?

Trong HTML5, <html>thẻ và <body>thẻ có thể được bỏ qua.

Đoạn mã sau sẽ xác thực là HTML5:

<!DOCTYPE html>  
<head>  
  <title>Page Title</title>  
</head>  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>

Tuy nhiên, chúng tôi không khuyên bạn nên bỏ qua <html>và <body>thẻ.

Phần <html>tử là tài liệu gốc. Đây là nơi được đề xuất để chỉ định ngôn ngữ trang:

<!DOCTYPE html>  
<html lang="en-US">

Khai báo một ngôn ngữ rất quan trọng đối với các ứng dụng trợ năng (trình đọc màn hình) và các công cụ tìm kiếm.

Bỏ sót <html>hoặc <body>có thể sập phần mềm DOM và XML.

Bỏ qua <body>có thể tạo ra lỗi trong các trình duyệt cũ hơn (IE9).

* Bỏ qua <head >?

Trong HTML5, thẻ <head> cũng có thể được bỏ qua.

Theo mặc định, trình duyệt sẽ thêm tất cả các yếu tố trước <body>vào một <head> yếu tố mặc định .

* Metadata

Phần <title>tử được yêu cầu trong HTML5. Làm cho tiêu đề có ý nghĩa nhất có thể:

<title>HTML5 Syntax and Coding Style</title>

Để đảm bảo giải thích chính xác và lập chỉ mục công cụ tìm kiếm chính xác, cả ngôn ngữ và mã hóa ký tự phải được xác định càng sớm càng tốt trong tài liệu:

<!DOCTYPE html>  
<html lang="en-US">  
<head>  
  <meta charset="UTF-8">  
  <title>HTML5 Syntax and Coding Style</title>  
</head>

* Đặt Chế độ xem

HTML5 đã giới thiệu một phương pháp để cho phép các nhà thiết kế web kiểm soát chế độ xem, thông qua <meta>thẻ.

Khung nhìn là khu vực hiển thị của người dùng trên trang web. Nó thay đổi theo thiết bị và sẽ nhỏ hơn trên điện thoại di động so với trên màn hình máy tính.

Bạn nên bao gồm <meta>yếu tố chế độ xem sau trong tất cả các trang web của mình:

<meta name="viewport" content="width=device-width, initial-scale=1.0">

Phần <meta>tử khung nhìn cung cấp cho hướng dẫn trình duyệt về cách kiểm soát kích thước và tỷ lệ của trang.

Phần width = width-width thiết lập chiều rộng của trang để theo chiều rộng màn hình của thiết bị (sẽ thay đổi tùy theo thiết bị).

Phần scale-scale = 1.0 đặt mức thu phóng ban đầu khi trang được trình duyệt tải lần đầu tiên.

Dưới đây là ví dụ về một trang web không có thẻ meta chế độ xem và cùng một trang web có thẻ meta chế độ xem:

* Nhận xét HTML

Bình luận ngắn nên được viết trên một dòng, như thế này:

<!-- This is a comment -->

Nhận xét kéo dài hơn một dòng, nên được viết như thế này:

<!--  
  This is a long comment example. This is a long comment example.  
  This is a long comment example. This is a long comment example.  
-->

Nhận xét dài sẽ dễ quan sát hơn nếu chúng được thụt vào hai khoảng trắng.

* Style sheets

Sử dụng cú pháp đơn giản để liên kết đến các biểu định kiểu (thuộc tính loại không cần thiết):

<link rel="stylesheet" href="styles.css">

Quy tắc ngắn có thể được viết nén, như thế này:

p.intro {font-family: Verdana; font-size: 16em;}

Các quy tắc dài nên được viết trên nhiều dòng:

body {  
  background-color: lightgrey;  
  font-family: "Arial Black", Helvetica, sans-serif;  
  font-size: 16em;  
  color: black;  
}

* Đặt khung mở trên cùng một dòng với bộ chọn
* Sử dụng một khoảng trắng trước dấu ngoặc mở
* Sử dụng hai không gian thụt
* Sử dụng dấu chấm phẩy sau mỗi cặp giá trị thuộc tính, bao gồm cả dấu cuối cùng
* Chỉ sử dụng dấu ngoặc kép quanh các giá trị nếu giá trị chứa khoảng trắng
* Đặt khung đóng trên một dòng mới, không có khoảng trắng ở đầu

Tránh các dòng trên 80 ký tự

* Đang tải JavaScript bằng HTML

Sử dụng cú pháp đơn giản để tải tập lệnh bên ngoài (thuộc tính type là không cần thiết):

<script src="myscript.js">

* Truy cập các phần tử HTML bằng JavaScript

Hậu quả của việc sử dụng các kiểu HTML "không gọn gàng" có thể dẫn đến lỗi JavaScript.

Hai câu lệnh JavaScript này sẽ tạo ra các kết quả khác nhau:

var obj = getElementById("Demo")  
var obj = getElementById("demo")

* Sử dụng tên tệp chữ thường

Một số máy chủ web (Apache, Unix) phân biệt chữ hoa chữ thường về tên tệp: "london.jpg" không thể được truy cập là "London.jpg".

Các máy chủ web khác (Microsoft, IIS) không phân biệt chữ hoa chữ thường: "london.jpg" có thể được truy cập là "London.jpg" hoặc "london.jpg".

Nếu bạn sử dụng kết hợp chữ hoa và chữ thường, bạn phải cực kỳ nhất quán.

Nếu bạn chuyển từ một trường hợp không nhạy cảm sang một máy chủ phân biệt chữ hoa chữ thường, ngay cả những lỗi nhỏ cũng sẽ phá vỡ trang web của bạn!

Để tránh những vấn đề này, luôn luôn sử dụng tên tệp chữ thường.

* Phần mở rộng tập tin

Các tệp HTML phải có phần mở rộng .html hoặc .htm .

Các tệp CSS nên có phần mở rộng .css .

Các tệp JavaScript nên có phần mở rộng .js

* Sự khác nhau giữa .htm và .html

Không có sự khác biệt giữa các phần mở rộng .htm và .html. Cả hai sẽ được coi là HTML bởi bất kỳ trình duyệt web hoặc máy chủ web.

Sự khác biệt là văn hóa:

.htm "ngửi" các hệ thống DOS đầu tiên trong đó hệ thống giới hạn các phần mở rộng chỉ còn 3 ký tự.

.html "ngửi" các hệ điều hành Unix không có giới hạn này.

* .Sự khác biệt kỹ thuật

Khi một URL không chỉ định tên tệp (như https://www.w3schools.com/css/), máy chủ sẽ trả về một tên tệp mặc định. Tên tệp mặc định phổ biến là index.html, index.htmlm, default.html và default.htmlm.

Nếu máy chủ của bạn chỉ được định cấu hình với "index.html" làm tên tệp mặc định, tệp của bạn phải được đặt tên là "index.html", chứ không phải "index.htmlm."

Tuy nhiên, các máy chủ có thể được cấu hình với nhiều tên tệp mặc định và thông thường bạn có thể thiết lập bao nhiêu tên tệp mặc định nếu cần.

Dù sao, phần mở rộng đầy đủ cho các tệp HTML là .html và không có lý do gì nó không nên được sử dụng

### HTML Graphics

#### HTML Canvas

* HTML Canvas là gì?

Phần <canvas>tử HTML được sử dụng để vẽ đồ họa, nhanh chóng, thông qua JavaScript.

Các <canvas>yếu tố chỉ là một container cho đồ họa. Bạn phải sử dụng JavaScript để thực sự vẽ đồ họa.

Canvas có một số phương pháp để vẽ đường dẫn, hộp, vòng tròn, văn bản và thêm hình ảnh.

* Hỗ trợ trình duyệt
* Ví dụ về Canvas

Canvas là một khu vực hình chữ nhật trên trang HTML. Theo mặc định, một khung vẽ không có viền và không có nội dung.

Đánh dấu trông như thế này:

<canvas id="myCanvas" width="200" height="100"></canvas>

Lưu ý: Luôn chỉ định một idthuộc tính (sẽ được đề cập trong tập lệnh) và a widthvà heightthuộc tính để xác định kích thước của khung vẽ. Để thêm một đường viền, sử dụng stylethuộc tính.

Dưới đây là một ví dụ về một khung vẽ cơ bản, trống rỗng:

<canvas id="myCanvas" width="200" height="100" style="border:1px solid #000000;">  
</canvas>

Vẽ đường thẳng

var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
ctx.moveTo(0, 0);  
ctx.lineTo(200, 100);  
ctx.stroke();

Vẽ một vòng tròn

var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
ctx.beginPath();  
ctx.arc(95, 50, 40, 0, 2 \* Math.PI);  
ctx.stroke();

Vẽ một văn bản

var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
ctx.font = "30px Arial";  
ctx.fillText("Hello World", 10, 50);

Stroke Text

var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
ctx.font = "30px Arial";  
ctx.strokeText("Hello World", 10, 50);

Vẽ Gradient tuyến tính

var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
// Create gradient  
var grd = ctx.createLinearGradient(0, 0, 200, 0);  
grd.addColorStop(0, "red");  
grd.addColorStop(1, "white");  
// Fill with gradient  
ctx.fillStyle = grd;  
ctx.fillRect(10, 10, 150, 80);

Vẽ Gradient tròn

var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
// Create gradient  
var grd = ctx.createRadialGradient(75, 50, 5, 90, 60, 100);  
grd.addColorStop(0, "red");  
grd.addColorStop(1, "white");  
// Fill with gradient  
ctx.fillStyle = grd;  
ctx.fillRect(10, 10, 150, 80);

Vẽ hình

var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
var img = document.getElementById("scream");  
ctx.drawImage(img, 10, 10);

#### HTML SVG

* SVG là gì?

SVG là viết tắt của Đồ họa vectơ có thể mở rộng

SVG được sử dụng để xác định đồ họa cho Web

SVG là một khuyến nghị của W3C

* Phần tử <svg> HTML

Phần <svg>tử HTML là một thùng chứa cho đồ họa SVG.

SVG có một số phương pháp để vẽ đường dẫn, hộp, vòng tròn, văn bản và hình ảnh đồ họa.

* Hỗ trợ trình duyệt

Các số trong bảng chỉ định phiên bản trình duyệt đầu tiên hỗ trợ đầy đủ <svg>phần tử.

* Vòng tròn SVG

<!DOCTYPE html>  
<html>  
<body>  
<svg width="100" height="100">  
  <circle cx="50" cy="50" r="40" stroke="green" stroke-width="4" fill="yellow" />  
</svg>  
</body>  
</html>

* Hình chữ nhật SVG

<svg width="400" height="100">  
  <rect width="400" height="100" style="fill:rgb(0,0,255);stroke-width:10;stroke:rgb(0,0,0)" />  
</svg>

* Hình chữ nhật tròn SVG

<svg width="400" height="180">  
  <rect x="50" y="20" rx="20" ry="20" width="150" height="150"  
  style="fill:red;stroke:black;stroke-width:5;opacity:0.5" />  
</svg>

* Ngôi sao SVG

<svg width="300" height="200">  
  <polygon points="100,10 40,198 190,78 10,78 160,198"  
  style="fill:lime;stroke:purple;stroke-width:5;fill-rule:evenodd;" />  
</svg>

* Logo SVG

<svg height="130" width="500">  
  <defs>  
    <linearGradient id="grad1" x1="0%" y1="0%" x2="100%" y2="0%">  
      <stop offset="0%" style="stop-color:rgb(255,255,0);stop-opacity:1" />  
      <stop offset="100%" style="stop-color:rgb(255,0,0);stop-opacity:1" />  
    </linearGradient>  
  </defs>  
  <ellipse cx="100" cy="70" rx="85" ry="55" fill="url(#grad1)" />  
  <text fill="#ffffff" font-size="45" font-family="Verdana" x="50" y="86">SVG</text>  
  Sorry, your browser does not support inline SVG.  
</svg>

* Sự khác biệt giữa SVG và Canvas

SVG là ngôn ngữ để mô tả đồ họa 2D bằng XML.

Canvas vẽ đồ họa 2D, nhanh chóng (bằng JavaScript).

SVG dựa trên XML, có nghĩa là mọi phần tử đều có sẵn trong DOMG DOM. Bạn có thể đính kèm các trình xử lý sự kiện JavaScript cho một phần tử.

Trong SVG, mỗi hình dạng được vẽ được ghi nhớ như một đối tượng. Nếu các thuộc tính của đối tượng SVG bị thay đổi, trình duyệt có thể tự động kết xuất lại hình dạng.

Canvas được hiển thị pixel theo pixel. Trong canvas, một khi đồ họa được vẽ, nó sẽ bị trình duyệt quên lãng. Nếu vị trí của nó phải được thay đổi, toàn bộ cảnh cần phải được vẽ lại, bao gồm bất kỳ đối tượng nào có thể được bao phủ bởi đồ họa.

* So sánh Canvas và SVG

Bảng bên dưới cho thấy một số khác biệt quan trọng giữa Canvas và SVG:



### HTML Media

#### HTML Media

Đa phương tiện trên web là âm thanh, âm nhạc, video, phim và hoạt hình.

Thẻ embed trong HTML

Đôi khi bạn cần thêm video hoặc nhạc vào trong trang web của mình. Cách đơn giản nhất để thêm video hay âm thanh vào site của bạn là bao nó trong một thẻ HTML gọi là <embed>.

Bạn cũng có thể sử dụng một thẻ <noembed> đối với các trình duyệt không nhận thẻ <embed>. Ví dụ, bạn có thể sử dụng thẻ <embed> để hiển thị một đoạn phim và sử dụng thẻ <noembed> để hiển thị các ảnh IPG nếu trình duyệt không hỗ trợ thẻ <embed>.

Ví dụ

Sau đây là một ví dụ đơn giản để chạy một file dạng midi được nhúng:

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Thẻ embed trong HTML**</title>**

**</head>**

**<body>**

**<embed** src="../html/yourfile.mid" width="100%" height="60" **>**

**<noembed><img** src="yourimage.gif" alt="Alternative Media" **></noembed>**

**</embed>**

**</body>**

**</html>**

Bạn có thể đặt bất kỳ file đa phương tiện nào trong thuộc tính src. Bạn hãy tự mình thực hành các kiểu đa phương tiện khác nhau.

Thuộc tính của thẻ <embed> trong HTML

Bảng dưới đây liệt kê danh sách các thuộc tính được sử dụng trong thẻ <embed>.

|  |  |
| --- | --- |
| Thuộc tính | Miêu tả |
| align | Căn chỉnh đối tượng: **center, left** hoặc **right**. |
| autostart | Thuộc tính logic này xác định có hay không đa phương tiện nên khởi động tự động. Giá trị là *true* hoặc *false*. |
| loop | Xác định nếu âm thanh sẽ được tiếp tục chơi (thiết lập loop là true), hay chơi trong một thời gian nhất định (một giá trị xác định), hoặc không (false). |
| playcount | Xác định lượng thời gian để âm thanh được chơi. Đây là tùy chọn thay thế cho tùy chọn *loop* nếu bạn đang sử dụng IE. |
| hidden | Xác định đối tượng đa phương tiện sẽ được hiển thị trên trang. Giá trị false nghĩa là không và true là có. |
| width | Độ rộng của đối tượng thể hiện bằng pixel. |
| height | Chiều cao của đối tượng thể hiện bằng pixel. |
| name | Tên được sử dụng để tham chiếu của đối tượng. |
| src | Địa chỉ URL của đối tượng được nhúng. |
| volume | Điều khiển âm lượng của âm thanh. Giá trị từ 0 (tắt) đến 100 (âm lượng to nhất). |

#### HTML Video

#### Các kiểu Video được hỗ trợ

Bạn có thể sử dụng các kiểu đa phương tiện khác nhau như Flash movies (.swf), AVI's (.avi), và MOV's (.mov) trong thẻ embed.

* .swf files - là các file được tạo bởi chương trình Macromedia's Flash.
* .wmv files - được tạo bởi Microsoft's Window's Media Video.
* .mov files - được tạo bởi định dạng Apple's Quick Time Movie.
* .mpeg files - được tạo bởi Moving Pictures Expert Group.

Vi du

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du the embed trong HTML**</title>**

**</head>**

**<body>**

**<embed** src="../html/yourfile.swf" width="200" height="200" **>**

**<noembed><img** src="yourimage.gif" alt="Alternative Media" **></noembed>**

**</embed>**

**</body>**

**</html>**

#### HTML Audio

Background Audio trong HTML

Bạn có thể sử dụng thẻ <bgsound> để chơi một bản soundtrack trong nền của trang web. Thẻ này chỉ được hỗ trợ bởi IE và hầu hết các trình duyệt khác bỏ qua thẻ này. Thẻ này tải về và chơi một tệp âm thanh khi tài liệu lưu trữ được tải về lần đầu tiên bởi người sử dụng và được hiển thị. Tệp âm thanh nền cũng sẽ chơi lại bất cứ khi nào người sử dụng tải lại trình duyệt.

Thẻ này chỉ có hai thuộc tính là loop và src. Cả hai thuộc tính này đều có ý nghĩa như trên.

Sau đây là ví dụ để chơi một tệp mini nhỏ:

Nó sẽ tạo ra một màn hình trắng. Thẻ này không hiển thị bất kỳ yếu tố nào và bị ẩn đi.

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**Vi du the embed trong HTML**</title>**

**</head>**

**<body>**

**<bgsound** src="../html/yourfile.mid"**>**

**<noembed><img** src="yourimage.gif" **></noembed>**

**</bgsound>**

**</body>**

**</html>**

IE có thể cũng chỉ điều khiển 3 định dạng âm thanh khác nhau: wav (định dạng tự nhiên cho PCs); au (định dạng tự nhiên cho Unix); và MIDI.

#### HTML Plug-ins

Thẻ object trong HTML

HTML 4 giới thiệu phần tử <object>, mà cung cấp giải pháp chung dựa trên các đối tượng. Phần tử này cho phép lập trình viên HTML xác định mọi thứ cần thiết bởi một đối tượng để trình bày trên Client.

Dưới đây là các ví dụ:

Ví dụ - 1

Bạn có thể nhúng một tài liệu HTML trong một tài liệu HTML như sau:

**<object** data="data/test.htm" type="text/html" width="300" height="200"**>**

  alt : **<a** href="data/test.htm"**>**test.htm**</a>**

**</object>**

Tại đây thì thuộc tính alt sẽ được sử dụng cho hình ảnh nếu trình duyệt không hỗ trợ thẻ object.

Ví dụ - 2

Bạn có thể nhúng một tài liệu PDF trong một tài liệu HTML như sau:

**<object** data="data/test.pdf" type="application/pdf" width="300" height="200"**>**

  alt : **<a** href="data/test.pdf"**>**test.htm**</a>**

**</object>**

Ví dụ - 3

Bạn có thể xác định một số tham số liên quan tới thẻ <param>. Dưới đây là ví dụ về một tệp dạng wam được nhúng:

**<object** data="data/test.wav" type="audio/x-wav" width="200" height="20"**>**

**<param** name="src" value="data/test.wav"**>**

**<param** name="autoplay" value="false"**>**

**<param** name="autoStart" value="0"**>**

  alt : **<a** href="data/test.wav"**>**test.wav**</a>**

**</object>**

Ví dụ - 4

Bạn có thể thêm một tài liệu flash như sau:

**<object** classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000" id="penguin"

   codebase="someplace/swflash.cab" width="200" height="300"**>**

**<param** name="movie" value="flash/penguin.swf" **/>**

**<param** name="quality" value="high" **/>**

**<img** src="penguin.jpg" width="200" height="300" alt="Penguin" **/>**

**</object>**

Ví dụ - 5

Bạn có thể thêm một mã Java vào trong tài liệu HTML như sau:

**<object** classid="clsid:8ad9c840-044e-11d1-b3e9-00805f499d93"

   width="200" height="200"**>**

**<param** name="code" value="applet.class"**>**

**</object>**

Thuộc tính classid xác định phiên bản của Java Plug-in được sử dụng. Bạn có thể sử dụng thuộc tính codebase để xác định nếu và cách tải JRE.

#### HTML YouTube

Phát video YouTube bằng HTML

Để phát video của bạn trên một trang web, hãy làm như sau:

* Tải video lên YouTube
* Ghi lại id video
* Xác định phần tử <iframe> trong trang web của bạn
* Để thuộc tính src trỏ đến URL video
* Sử dụng các thuộc tính chiều rộng và chiều cao để chỉ định kích thước của trình phát
* Thêm bất kỳ tham số nào khác vào URL (xem bên dưới)
* Tự động phát YouTube
* Bạn có thể để video của mình bắt đầu phát tự động khi người dùng truy cập trang đó bằng cách thêm một tham số đơn giản vào URL YouTube của bạn.
* Lưu ý: Hãy cân nhắc cẩn thận khi quyết định tự động phát video của bạn. Tự động bắt đầu một video có thể gây khó chịu cho khách truy cập của bạn và cuối cùng gây ra nhiều tác hại hơn là tốt.
* Giá trị 0 (mặc định): Video sẽ không tự động phát khi trình phát tải.
* Giá trị 1: Video sẽ tự động phát khi trình phát tải.

### HTML APIs

#### HTML Geolocation

Xác định vị trí của người dùng

API vị trí địa lý HTML được sử dụng để có được vị trí địa lý của người dùng.

Vì điều này có thể ảnh hưởng đến quyền riêng tư, vị trí không có sẵn trừ khi người dùng chấp thuận.

Lưu ý: Định vị địa lý là chính xác nhất cho các thiết bị có GPS, như điện thoại thông minh.

Lưu ý: Kể từ Chrome 50, API vị trí địa lý sẽ chỉ hoạt động trên các bối cảnh an toàn như HTTPS. Nếu trang web của bạn được lưu trữ trên một nguồn gốc không an toàn (như HTTP), các yêu cầu để có được vị trí người dùng sẽ không còn hoạt động.

Sử dụng định vị địa lý HTML

Các getCurrentPosition()phương pháp được sử dụng để trả vị trí của người dùng.

Ví dụ dưới đây trả về vĩ độ và kinh độ của vị trí người dùng:

<script>

var x = document.getElementById("demo");

function getLocation() {

  if (navigator.geolocation) {

    navigator.geolocation.getCurrentPosition(showPosition);

  } else {

    x.innerHTML = "Geolocation is not supported by this browser.";

  }

}

function showPosition(position) {

  x.innerHTML = "Latitude: " + position.coords.latitude +

  "<br>Longitude: " + position.coords.longitude;

}

</script>

Giải thích:

* Kiểm tra nếu Định vị địa lý được hỗ trợ
* Nếu được hỗ trợ, hãy chạy phương thức getCienPocation (). Nếu không, hiển thị một thông báo cho người dùng
* Nếu phương thức getCienP vị trí () thành công, nó trả về một đối tượng tọa độ cho hàm được chỉ định trong tham số (showPocation)
* Hàm showPocation () xuất ra Vĩ độ và Kinh độ

Ví dụ trên là một tập lệnh Định vị địa lý rất cơ bản, không có xử lý lỗi.

Xử lý lỗi và từ chối

Tham số thứ hai của getCurrentPosition()phương thức được sử dụng để xử lý lỗi. Nó chỉ định một chức năng để chạy nếu không lấy được vị trí của người dùng:

Thí dụ

function showError(error) {

  switch(error.code) {

    case error.PERMISSION\_DENIED:

      x.innerHTML = "User denied the request for Geolocation."

      break;

    case error.POSITION\_UNAVAILABLE:

      x.innerHTML = "Location information is unavailable."

      break;

    case error.TIMEOUT:

      x.innerHTML = "The request to get user location timed out."

      break;

    case error.UNKNOWN\_ERROR:

      x.innerHTML = "An unknown error occurred."

      break;

  }

}

Hiển thị kết quả trong bản đồ

Để hiển thị kết quả trong bản đồ, bạn cần truy cập vào dịch vụ bản đồ, như Google Maps.

Trong ví dụ bên dưới, vĩ độ và kinh độ được trả về được sử dụng để hiển thị vị trí trong Google Map (sử dụng hình ảnh tĩnh):

Thí dụ

function showPosition(position) {

  var latlon = position.coords.latitude + "," + position.coords.longitude;

  var img\_url = "https://maps.googleapis.com/maps/api/staticmap?center=

  "+latlon+"&zoom=14&size=400x300&sensor=false&key=YOUR\_KEY";

  document.getElementById("mapholder").innerHTML = "<img src='"+img\_url+"'**>**";

}

Thông tin cụ thể theo địa điểm

Trang này đã trình bày cách hiển thị vị trí của người dùng trên bản đồ.

Định vị địa lý cũng rất hữu ích cho thông tin cụ thể theo vị trí, như:

* Thông tin địa phương cập nhật
* Hiển thị Điểm ưa thích gần người dùng
* Điều hướng từng chặng (GPS)

Phương thức getCienPocation () - Trả về dữ liệu

Các getCurrentPosition()phương thức trả về một đối tượng trên thành công. Các thuộc tính vĩ độ, kinh độ và độ chính xác luôn được trả về. Các thuộc tính khác được trả lại nếu có:

Property Returns

coords.latitude The latitude as a decimal number (always returned)

coords.longitude The longitude as a decimal number (always returned)

coords.accuracy The accuracy of position (always returned)

coords.altitude The altitude in meters above the mean sea level (returned if available)

coords.altitudeAccuracy The altitude accuracy of position (returned if available)

coords.heading The heading as degrees clockwise from North (returned if available)

coords.speed The speed in meters per second (returned if available)

timestamp The date/time of the response (returned if available)

Đối tượng định vị địa lý - Các phương pháp thú vị khác

Đối tượng Định vị địa lý cũng có các phương thức thú vị khác:

watchPosition() - Trả về vị trí hiện tại của người dùng và tiếp tục trả về vị trí được cập nhật khi người dùng di chuyển (như GPS trong xe hơi).

clearWatch()- Dừng watchPosition()phương thức.

Ví dụ dưới đây cho thấy watchPosition()phương pháp. Bạn cần một thiết bị GPS chính xác để kiểm tra điều này (như điện thoại thông minh):

<script>

var x = document.getElementById("demo");

function getLocation() {

  if (navigator.geolocation) {

    navigator.geolocation.watchPosition(showPosition);

  } else {

    x.innerHTML = "Geolocation is not supported by this browser.";

  }

}

function showPosition(position) {

  x.innerHTML = "Latitude: " + position.coords.latitude +

  "<br>Longitude: " + position.coords.longitude;

}

</script>

#### HTML Drag/Drop

Kéo và thả là một tính năng rất phổ biến. Đó là khi bạn "chộp" một vật thể và kéo nó đến một vị trí khác.

Trong HTML5, kéo và thả là một phần của tiêu chuẩn: Bất kỳ yếu tố nào cũng có thể kéo được.

Ví dụ kéo và thả HTML

Ví dụ dưới đây là một ví dụ kéo và thả đơn giản:

<!DOCTYPE HTML**>**

**<html>**

**<head>**

**<script>**

function allowDrop(ev) {

  ev.preventDefault();

}

function drag(ev) {

  ev.dataTransfer.setData("text", ev.target.id);

}

function drop(ev) {

  ev.preventDefault();

  var data = ev.dataTransfer.getData("text");

  ev.target.appendChild(document.getElementById(data));

}

**</script>**

**</head>**

**<body>**

**<div** id="div1" ondrop="drop(event)" ondragover="allowDrop(event)"**></div>**

**<img** id="drag1" src="img\_logo.gif" draggable="true" ondragstart="drag(event)" width="336" height="69"**>**

**</body>**

</html>

Tạo một phần tử có thể kéo

Trước hết: Để làm cho một phần tử có thể kéo được, hãy đặt draggablethuộc tính thành true:

<img draggable="true">

Kéo gì - ondragstart và setData ()

Sau đó, chỉ định những gì sẽ xảy ra khi phần tử được kéo.

Trong ví dụ trên, ondragstartthuộc tính gọi một hàm, kéo (sự kiện), chỉ định dữ liệu nào sẽ được kéo.

Các dataTransfer.setData()phương pháp thiết lập kiểu dữ liệu và giá trị của dữ liệu kéo:

function drag(ev) {

ev.dataTransfer.setData("text", ev.target.id);

}

Trong trường hợp này, kiểu dữ liệu là "văn bản" và giá trị là id của phần tử có thể kéo ("drag1").

Nơi để thả - ondragover

Sự ondragoverkiện chỉ định nơi dữ liệu được kéo có thể được thả.

Theo mặc định, dữ liệu / phần tử không thể được loại bỏ trong các phần tử khác. Để cho phép thả, chúng ta phải ngăn việc xử lý phần tử mặc định.

Điều này được thực hiện bằng cách gọi event.preventDefault()phương thức cho sự kiện ondragover:

event.preventDefault()

Làm thả - ondrop

Khi dữ liệu kéo được thả, một sự kiện thả xảy ra.

Trong ví dụ trên, thuộc tính ondrop gọi một hàm, drop (event):

function drop(ev) {

ev.preventDefault();

var data = ev.dataTransfer.getData("text");

ev.target.appendChild(document.getElementById(data));

}

Mã giải thích:

* Gọi notifyDefault () để ngăn việc xử lý dữ liệu mặc định của trình duyệt (mặc định là mở dưới dạng liên kết khi thả)
* Lấy dữ liệu được kéo bằng phương thức dataTransfer.getData (). Phương thức này sẽ trả về bất kỳ dữ liệu nào được đặt cùng loại trong phương thức setData ()
* Dữ liệu được kéo là id của phần tử được kéo ("drag1")
* Nối phần tử được kéo vào phần tử drop

#### HTML Web Storage

Lưu trữ web HTML là gì?

Với lưu trữ web, các ứng dụng web có thể lưu trữ dữ liệu cục bộ trong trình duyệt của người dùng.

Trước HTML5, dữ liệu ứng dụng phải được lưu trữ trong cookie, được bao gồm trong mọi yêu cầu máy chủ. Lưu trữ web an toàn hơn và một lượng lớn dữ liệu có thể được lưu trữ cục bộ mà không ảnh hưởng đến hiệu suất trang web.

Không giống như cookie, giới hạn lưu trữ lớn hơn nhiều (ít nhất 5 MB) và thông tin không bao giờ được chuyển đến máy chủ.

Lưu trữ web là mỗi nguồn gốc (mỗi tên miền và giao thức). Tất cả các trang, từ một nguồn gốc, có thể lưu trữ và truy cập cùng một dữ liệu.

Đối tượng lưu trữ web HTML

Lưu trữ web HTML cung cấp hai đối tượng để lưu trữ dữ liệu trên máy khách:

window.localStorage - lưu trữ dữ liệu không có ngày hết hạn

window.sessionStorage - lưu trữ dữ liệu cho một phiên (dữ liệu bị mất khi tab trình duyệt bị đóng)

Trước khi sử dụng lưu trữ web, hãy kiểm tra hỗ trợ trình duyệt cho localStorage và sessionStorage:

if (typeof(Storage) !== "undefined") {

// Code for localStorage/sessionStorage.

} else {

// Sorry! No Web Storage support..

}

Đối tượng LocalStorage

Đối tượng localStorage lưu trữ dữ liệu không có ngày hết hạn. Dữ liệu sẽ không bị xóa khi đóng trình duyệt và sẽ có sẵn vào ngày, tuần hoặc năm tiếp theo.

Thí dụ

// Store

localStorage.setItem("lastname", "Smith");

// Retrieve

document.getElementById("result").innerHTML = localStorage.getItem("lastname");

Ví dụ giải thích:

* Tạo một cặp tên / giá trị localStorage với name = "lastname" và value = "Smith"
* Lấy giá trị của "họ" và chèn nó vào phần tử với id = "result"

Ví dụ trên cũng có thể được viết như thế này:

// Store

localStorage.lastname = "Smith";

// Retrieve

document.getElementById("result").innerHTML = localStorage.lastname;

Cú pháp để xóa mục "Lastname" localStorage như sau:

localStorage.removeItem("lastname");

Lưu ý: Các cặp tên / giá trị luôn được lưu dưới dạng chuỗi. Hãy nhớ chuyển đổi chúng sang định dạng khác khi cần thiết!

Ví dụ sau đây đếm số lần người dùng đã nhấp vào nút. Trong mã này, chuỗi giá trị được chuyển đổi thành một số để có thể tăng bộ đếm:

if (localStorage.clickcount) {

  localStorage.clickcount = Number(localStorage.clickcount) + 1;

} else {

  localStorage.clickcount = 1;

}

document.getElementById("result").innerHTML = "You have clicked the button " +

localStorage.clickcount + " time(s).";

Đối tượng sessionStorage

Đối sessionStoragetượng bằng với đối tượng localStorage, ngoại trừ việc nó lưu trữ dữ liệu chỉ trong một phiên. Dữ liệu sẽ bị xóa khi người dùng đóng tab trình duyệt cụ thể.

Ví dụ sau đây đếm số lần người dùng đã nhấp vào nút, trong phiên hiện tại:

if (sessionStorage.clickcount) {

  sessionStorage.clickcount = Number(sessionStorage.clickcount) + 1;

} else {

  sessionStorage.clickcount = 1;

}

document.getElementById("result").innerHTML = "You have clicked the button " +

sessionStorage.clickcount + " time(s) in this session.";

#### HTML Web Workers

Web workers:

Web workers (Các trình làm việc trên nền web) khắc phục các vấn đề gây ra bởi tính đồng thời. Web workers là câu trả lời của họ HTML5 cho vấn đề đơn-luồng của JavaScript: Chúng chạy các quá trình trên một luồng riêng biệt từ trang chính, bảo vệ trang với các chức năng chính, chẳng hạn như duy trì một giao diện người dùng ổn định.

Một web worker là một tệp JavaScript được nạp và được thực hiện trong nền. Các web worker này cho phép bạn tải một tệp JavaScript động, và sau đó thực hiện một kịch bản lệnh bằng cách sử dụng một quá trình nền mà không ảnh hưởng đến giao diện người dùng. Các web worker có quyền truy cập hạn chế và chỉ được phép truyền qua các chuỗi. Vì web worker không sử dụng luồng giao diện người dùng của trình duyệt, nên chúng không được phép truy cập vào DOM. Các worker có thể sử dụng cả hai tham chiếu self và this cho phạm vi chung của worker. Truyền dẫn của worker và trang cha mẹ đạt được bằng cách sử dụng một mô hình sự kiện và phương thức postMessage().

Vì Web workers có cách hoạt động đa luồng, nên chúng chỉ có thể truy cập một tập con của các tính năng của JavaScript. Web workers có thể:

* Truy cập đối tượng navigator: Sử dụng đối tượng vị trí chỉ-đọc.
* Thực hiện XMLHttpRequest để gửi các yêu cầu HTTP hoặc HTTPS.
* Thiết lập một thời gian hoặc khoảng thời gian cho một hoạt động bằng cách sử dụng setTimeout()/clearTimeout() và setInterval()/clearInterval().
* Truy cập bộ nhớ ứng dụng.
* Nhập kịch bản lệnh bên ngoài bằng cách sử dụng phương thức importScripts().
* Sinh ra các Web worker khác (worker con - subworker - phải có cùng nguồn gốc như trang chính và được đặt trong cùng một vị trí như worker mẹ).

Có hai kiểu web workers:

* các worker chuyên dụng.
* các worker chia sẻ.

Web worker chuyên dụng:

Một web worker chuyên dụng được liên kết đến kịch bản lệnh tạo ra nó, và nó có thể truyền thông với các thành phần worker hoặc trình duyệt khác. Tuy nhiên, nó không thể truyền thông với các DOM.

Một web worker chuyên dụng được tạo ra bằng cách chuyển một tên tệp JavaScript tới một cá thể worker mới. Bạn tạo ra một worker mới khi sử dụng hàm tạo Worker() bằng cách chỉ rõ địa chỉ URI của kịch bản lệnh đang thực hiện của worker. Để tạo ra một worker chuyên dụng, hãy nhập mã được hiển thị dưới đây, mã này tạo ra một đối tượng Worker chuyên dụng mới Worker:

var worker = new Worker('worker.js');

Các web worker chia sẻ:

Các web worker chia sẻ, giống như các worker chuyên dụng, không thể truy cập vào DOM và chỉ có quyền truy cập hạn chế vào các đặc tính cửa sổ. Các web worker chia sẻ chỉ có thể truyền thông với các web worker chia sẻ khác của cùng một miền. Các worker này được tạo ra bằng cách chuyển một tên JavaScript tới một cá thể worker chia sẻ mới..

Các kịch bản lệnh trang có thể truyền thông với các web worker chia sẻ. Tuy nhiên, không giống như các web worker chuyên dụng, bạn truyền thông bằng cách sử dụng một đối tượng port (cổng) và gán cho một trình xử lý sự kiện thông báo. Ngoài ra, bạn phải gọi phương thức start() của port trước khi sử dụng phương thức postMessage() đầu tiên.

Dựa vào việc nhận được thông báo đầu tiên bằng kịch bản lệnh web worker, web worker chia sẻ gán một trình xử lý sự kiện cho cổng hoạt động. Nói chung, trình xử lý sẽ chạy phương thức postMessage() riêng của nó để trả về một thông báo tới mã đang gọi, và sau đó phương thức start() của port tạo một quá trình thông báo cho phép.

Để tạo ra một web worker chia sẻ, bạn phải tạo một đối tượng SharedWorker thay cho đối tượng Worker. Đoạn mã sau đây cho thấy cách tạo một đối tượng SharedWorker mới:

var worker = new SharedWorker('worker.js');

#### HTML SSE

HTML5 Server-Sent Events là gì?

HTML5 Server-Sent Events (SSE) hay còn được biết đến với tên gọi HTML5 Event Source là một Web Api cho phép kết nối thời gian thực giữa server và client (browser). Tuy nhiên, khác với WebSockets, SSE chỉ có 1 chiều dữ liệu, đó là từ server đến client (chỉ có server gửi dữ liệu đến client).

Tại sao dùng SSE khi đã có Web Sockets

Ưu điểm của SSE đó là API đơn giản, dễ thực hiện, không yêu cầu việc setup và implement phức tạp như Web sockets. SSE hoạt động với giao thức HTTP thông thường, không yêu cầu một Web socket server mới hay các giao thức đặc biệt khác.

Trong thực tế, không phải ứng dụng web real time nào cũng cần sự tương tác 2 chiều giữa client và server, trong trường hợp này việc sử dụng SSE là dễ dàng hơn rất nhiều. Có thể kế đến một số trường hợp như:

* Ứng dụng dashboard, đồ thị chart
* News feeds
* Ứng dụng theo dõi hoạt động của server
* Ứng dụng có dữ liệu thay đổi nhanh chóng như giá cả, thị trường, tình trạng hàng hóa, thông tin các trận đấu thể thao,...

Một ưu điểm nữa của SSE đó là, client có thể tự động kết nối lại server nếu bị mất kết nối, thời gian timeout trước khi client kết nối lại có thể tùy chỉnh được từ server. Khi thực hiện kết nối lại, client sẽ tự động gửi ID của event cuối cùng nó nhận được, nhờ đó server có thể biết được client đã bị bỏ qua bao nhiêu message để gửi lại những message đó.

SSE hoạt động với hầu hết các trình duyệt, ngoại trừ IE. Không biết trong tương lai IE có support hay nó có còn tồn tại hay không , tuy nhiên, chúng ta vẫn có thể sử dụng các polyfills để support đầy đủ tất cả các trình duyệt.

### HTML Examples

<!DOCTYPE html>  
<html>  
<body>  
<h1>Đây là tiêu đề</h1>  
<p>Đây là đoạn văn bản.</p>

<a href=”http://google.com”>Đây là link</a>

<img src=”images1.png” alt=”anhmau” width=”104px” height=”142px”>  
</body>  
</html>

### HTML References

|  |  |
| --- | --- |
| Thẻ | Mô tả |
| [<!--...-->](https://www.w3schools.com/tags/tag_comment.asp) | Định nghĩa một comment |
| [<!DOCTYPE>](https://www.w3schools.com/tags/tag_doctype.asp) | Định nghĩa loại tài liệu |
| [<a>](https://www.w3schools.com/tags/tag_a.asp) | Định nghĩa một siêu liên kết |
| [<abbr>](https://www.w3schools.com/tags/tag_abbr.asp) | Định nghĩa từ viết tắt hoặc tóm tắt một nội dung nào đó. |
| [<acronym>](https://www.w3schools.com/tags/tag_acronym.asp) | Không hỗ trợ trong HTML5. Thay thế sử dụng <abbr>.  Định nghĩa một từ viết tắt. |
| [<address>](https://www.w3schools.com/tags/tag_address.asp) | Định nghĩa thông tin liên lạc cho các tác giả/ chủ sở hữu của một tài liệu. |
| [<applet>](https://www.w3schools.com/tags/tag_applet.asp) | Không hỗ trợ trong HTML5. Thay thế sử dụng <embed> hoặc <object>.  Định nghĩa nhúng một applet. |
| [<area>](https://www.w3schools.com/tags/tag_area.asp) | Định nghĩa vùng bên trong của một ánh xạ hình ảnh. |
| [<article>](https://www.w3schools.com/tags/tag_article.asp) | Định nghĩa một bài báo |
| [<aside>](https://www.w3schools.com/tags/tag_aside.asp) | Định nghĩa nội dung nằm bên cạnh nội dung của trang |
| [<audio>](https://www.w3schools.com/tags/tag_audio.asp) | Định nghĩa một nội dung âm thanh. |
| [<b>](https://www.w3schools.com/tags/tag_b.asp) | Định nghĩa chữ in đậm. |
| [<base>](https://www.w3schools.com/tags/tag_base.asp) | Chỉ định URL cơ sở/target cho tất cả các URL tương đối trong một tài liệu |
| [<basefont>](https://www.w3schools.com/tags/tag_basefont.asp) | Không hỗ trợ trong HTML5. Sử dụng CSS để thay thế.  Chỉ định màu mặc định, kích thước và phông chữ cho tất cả phần văn bản trong tài liệu. |
| [<bdi>](https://www.w3schools.com/tags/tag_bdi.asp) | Tách biệt một phần của văn bản có thể được định dạng theo một hướng khác nhau từ văn bản khác bên ngoài nó |
| [<bdo>](https://www.w3schools.com/tags/tag_bdo.asp) | Ghi đè hướng văn bản hiện hành |
| [<big>](https://www.w3schools.com/tags/tag_big.asp) | Không hỗ trợ trong HTML5. Dùng CSS để thay thế.  Xác định văn bản lớn. |
| [<blockquote>](https://www.w3schools.com/tags/tag_blockquote.asp) | Định nghĩa một phần được trích dẫn từ nguồn khác |
| [<body>](https://www.w3schools.com/tags/tag_body.asp) | Định nghĩa than của tài liệu |
| [<br>](https://www.w3schools.com/tags/tag_br.asp) | Định nghĩa một ngắt dòng đơn |
| [<button>](https://www.w3schools.com/tags/tag_button.asp) | Định nghĩa một nút bấm |
| [<canvas>](https://www.w3schools.com/tags/tag_canvas.asp) | Được sử dụng để vẽ đồ họa, thông qua một ngôn ngữ script (thường là JavaScript) |
| [<caption>](https://www.w3schools.com/tags/tag_caption.asp) | Định nghĩa một chú thích trong bảng |
| [<center>](https://www.w3schools.com/tags/tag_center.asp) | Không hỗ trợ trong HTML5. Sử dụng CSS để thay thế.  Xác định văn bản làm trung tâm. |
| [<cite>](https://www.w3schools.com/tags/tag_cite.asp) | Định nghĩa nội dung được trích dẫn như một quyển sách, bài báo, một video,… |
| [<code>](https://www.w3schools.com/tags/tag_code.asp) | Định nghĩa một đoạn mã máy tính |
| [<col>](https://www.w3schools.com/tags/tag_col.asp) | Chỉ định thuộc tính cột cho mỗi cột trong một <colgroup> |
| [<colgroup>](https://www.w3schools.com/tags/tag_colgroup.asp) | Chỉ định một nhóm gồm một hoặc nhiều cột trong một bảng để định dạng |
| [<datalist>](https://www.w3schools.com/tags/tag_datalist.asp) | Định nghĩa một danh sách các giá trị tùy chọn được định nghĩa trước cho phần tử <input> |
| [<dd>](https://www.w3schools.com/tags/tag_dd.asp) | Được dùng để mô tả một từ khóa/ giá trị trong một danh sách mô tả |
| [<del>](https://www.w3schools.com/tags/tag_del.asp) | Định nghĩa văn bản đã bị xóa từ một tài liệu |
| [<details>](https://www.w3schools.com/tags/tag_details.asp) | Xác định các chi tiết khác mà người dùng có thể xem hoặc ẩn. |
| [<dfn>](https://www.w3schools.com/tags/tag_dfn.asp) | Mô tả định nghĩa của một từ |
| [<dialog>](https://www.w3schools.com/tags/tag_dialog.asp) | Định nghĩa một hộp thoại hoặc một cửa sổ |
| [<dir>](https://www.w3schools.com/tags/tag_dir.asp) | Không hỗ trợ trong HTML5. Sử dụng <ul> để thay thế.  Định nghĩa một danh sách thư mục. |
| [<div>](https://www.w3schools.com/tags/tag_div.asp) | Định nghĩa một phần trong tài liệu. |
| [<dl>](https://www.w3schools.com/tags/tag_dl.asp) | Định nghĩa một danh sách mô tả |
| [<dt>](https://www.w3schools.com/tags/tag_dt.asp) | Định nghĩa một từ khóa/tên trong danh sách mô tả. |
| [<em>](https://www.w3schools.com/tags/tag_em.asp) | Định nghĩa nhấn mạnh văn bản |
| [<embed>](https://www.w3schools.com/tags/tag_embed.asp) | Định nghĩa một container chứa một ứng dụng mở rộng(không phải HTML) |
| [<fieldset>](https://www.w3schools.com/tags/tag_fieldset.asp) | Nhóm lên nhóm các phần tử liên quan trong form |
| [<figcaption>](https://www.w3schools.com/tags/tag_figcaption.asp) | Định nghĩa một chú thích cho thẻ <figure> |
| [<figure>](https://www.w3schools.com/tags/tag_figure.asp) | Chỉ định nội dung độc lập |
| [<font>](https://www.w3schools.com/tags/tag_font.asp) | Không hỗ trợ trong HTML5. Sử dụng CSS để thay thế.  Xác định phông chữ, màu sắc , kích thước cho văn bản |
| [<footer>](https://www.w3schools.com/tags/tag_footer.asp) | Định nghĩa phần footer cho một tài liệu hoặc một đoạn |
| [<form>](https://www.w3schools.com/tags/tag_form.asp) | Định nghĩa một form HTML cho người dùng nhập vào. |
| [<frame>](https://www.w3schools.com/tags/tag_frame.asp) | Không hỗ trợ trong HTML5.  Định nghĩa một cửa sổ (khung) trong một khung |
| [<frameset>](https://www.w3schools.com/tags/tag_frameset.asp) | Không hỗ trợ trong HTML5.  Định nghĩa một tập hợp các frame |
| [<h1> to <h6>](https://www.w3schools.com/tags/tag_hn.asp) | Định nghĩa tiêu đề HTML |
| [<head>](https://www.w3schools.com/tags/tag_head.asp) | Xác định các thông tin về tài liệu |
| [<header>](https://www.w3schools.com/tags/tag_header.asp) | Định nghĩa một tiêu đề cho một tài liệu hoặc một phần |
| [<hr>](https://www.w3schools.com/tags/tag_hr.asp) | Sử dụng để phân tách các phần nội dung trong HTML |
| [<html>](https://www.w3schools.com/tags/tag_html.asp) | Định nghĩa gốc của một tài liệu HTML |
| [<i>](https://www.w3schools.com/tags/tag_i.asp) | Định nghĩa một phần chữ nghiêng cho văn bản |
| [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp) | Định nghĩa một khung nội tuyến. |
| [<img>](https://www.w3schools.com/tags/tag_img.asp) | Định nghĩa một hình ảnh |
| [<input>](https://www.w3schools.com/tags/tag_input.asp) | Định nghĩa một control nhập dữ liệu |
| [<ins>](https://www.w3schools.com/tags/tag_ins.asp) | Định nghĩa một đoạn văn bản đã được chèn thêm thay thế vào tài liệu |
| [<kbd>](https://www.w3schools.com/tags/tag_kbd.asp) | Là một thẻ dùng để nhấn mạnh, dùng để định nghĩa đầu vào bàn phím |
| [<label>](https://www.w3schools.com/tags/tag_label.asp) | Định nghĩa một nhãn cho một phần tử <input> |
| [<legend>](https://www.w3schools.com/tags/tag_legend.asp) | Định nghĩa một chú thích cho một phần tử <fieldset> |
| [<li>](https://www.w3schools.com/tags/tag_li.asp) | Định nghĩa một danh sách |
| [<link>](https://www.w3schools.com/tags/tag_link.asp) | Xác định mối quan hệ giữa một tài liệu và một nguồn lực bên ngoài (thường được sử dụng để liên kết đến style sheet) |
| [<main>](https://www.w3schools.com/tags/tag_main.asp) | Chỉ định các nội dung chính của một tài liệu |
| [<map>](https://www.w3schools.com/tags/tag_map.asp) | Định nghĩa một ánh xạ hình ảnh ở phía client |
| [<mark>](https://www.w3schools.com/tags/tag_mark.asp) | Định nghĩa đánh dấu/làm nổi bật văn bản |
| [<meta>](https://www.w3schools.com/tags/tag_meta.asp) | Định nghĩa siêu dữ liệu về một tài liệu HTML |
| [<meter>](https://www.w3schools.com/tags/tag_meter.asp) | Định nghĩa một thước đo lường với một khoảng đã biết |
| [<nav>](https://www.w3schools.com/tags/tag_nav.asp) | Định nghĩa liên kết điều hướng |
| [<noframes>](https://www.w3schools.com/tags/tag_noframes.asp) | Không hỗ trợ trong HTML5.  Định nghĩa một nội dung thay thế cho người dùng nếu trình duyệt không hỗ trợ khung nhìn. |
| [<noscript>](https://www.w3schools.com/tags/tag_noscript.asp) | Định nghĩa một nội dung thay thế cho người dùng mà trình duyệt phía client không hỗ trợ script |
| [<object>](https://www.w3schools.com/tags/tag_object.asp) | Định nghĩa một đối tượng nhúng vào trang |
| [<ol>](https://www.w3schools.com/tags/tag_ol.asp) | Định nghĩa một danh sách có sắp xếp |
| [<optgroup>](https://www.w3schools.com/tags/tag_optgroup.asp) | Định nghĩa một nhóm các tùy chọn liên quan trong một danh sách drop-down |
| [<option>](https://www.w3schools.com/tags/tag_option.asp) | Định nghĩa một lựa chọn trong danh sách drop-down |
| [<output>](https://www.w3schools.com/tags/tag_output.asp) | Xác định kết quả của một phép tính |
| [<p>](https://www.w3schools.com/tags/tag_p.asp) | Định nghĩa một đoạn văn bản |
| [<param>](https://www.w3schools.com/tags/tag_param.asp) | Định nghĩa một tham số cho một đối tượng |
| [<picture>](https://www.w3schools.com/tags/tag_picture.asp) | Xác định một vùng chứa cho nhiều tài nguyên hình ảnh |
| [<pre>](https://www.w3schools.com/tags/tag_pre.asp) | Định nghĩa văn bản định dạng sẵn |
| [<progress>](https://www.w3schools.com/tags/tag_progress.asp) | Miêu tả quá trình xử lý công việc bằng một thanh progressbar, thường được két hợp với code Javascript |
| [<q>](https://www.w3schools.com/tags/tag_q.asp) | Định nghĩa dấu nháy kép xung quanh phần văn bản |
| [<rp>](https://www.w3schools.com/tags/tag_rp.asp) | Định nghĩa những gì thể hiện trong các trình duyệt không hỗ trợ các chú thích |
| [<rt>](https://www.w3schools.com/tags/tag_rt.asp) | Định nghĩa diễn giải/phát âm của các ký tự (đối với kiểu chữ Đông Á) |
| [<ruby>](https://www.w3schools.com/tags/tag_ruby.asp) | Định nghĩa một chú thích (đối với kiểu chữ Đông Á) |
| [<s>](https://www.w3schools.com/tags/tag_s.asp) | Định nghĩa văn bản đó không còn đúng |
| [<samp>](https://www.w3schools.com/tags/tag_samp.asp) | Hiển thị kết quả đầu ra của một chương trình máy tính |
| [<script>](https://www.w3schools.com/tags/tag_script.asp) | Định nghĩa một kịch bản phía máy khách |
| [<section>](https://www.w3schools.com/tags/tag_section.asp) | Định nghĩa một phần trong một tài liệu |
| [<select>](https://www.w3schools.com/tags/tag_select.asp) | Định nghĩa một danh sách drop-down |
| [<small>](https://www.w3schools.com/tags/tag_small.asp) | Định nghĩa văn bản chữ nhỏ |
| [<source>](https://www.w3schools.com/tags/tag_source.asp) | Định nghĩa các nguồn đa phương tiện cho các phần tử đa phương tiện như <video> và <audio> |
| [<span>](https://www.w3schools.com/tags/tag_span.asp) | Định nghĩa một phần trong một tài liệu |
| [<strike>](https://www.w3schools.com/tags/tag_strike.asp) | Không hỗ trợ trong HTML5. Sử dụng <del> hoặc <s> thay thế.  Định nghĩa chữ gạch ngang |
| [<strong>](https://www.w3schools.com/tags/tag_strong.asp) | Định nghĩa văn bản in đậm, ý nghĩa nhấn mạnh độ quan trọng |
| [<style>](https://www.w3schools.com/tags/tag_style.asp) | Định nghĩa kiểu cho một tài liệu |
| [<sub>](https://www.w3schools.com/tags/tag_sub.asp) | Định nghĩa văn bản giống như chỉ số dưới |
| [<summary>](https://www.w3schools.com/tags/tag_summary.asp) | Hiển thị tiêu đề cho phần tử <detail> |
| [<sup>](https://www.w3schools.com/tags/tag_sup.asp) | Hiển thi văn bản giống số mũ |
| [<svg>](https://www.w3schools.com/tags/tag_svg.asp) | Xác định một vùng chứa cho đồ họa SVG |
| [<table>](https://www.w3schools.com/tags/tag_table.asp) | Định nghĩa một bảng |
| [<tbody>](https://www.w3schools.com/tags/tag_tbody.asp) | Nhóm các nội dung trong một bảng |
| [<td>](https://www.w3schools.com/tags/tag_td.asp) | Định nghĩa một ô trong một bảng |
| [<template>](https://www.w3schools.com/tags/tag_template.asp) | Xác định một mẫu |
| [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Xác định một ô nhập dữ liệu văn bản có nhiều dòng |
| [<tfoot>](https://www.w3schools.com/tags/tag_tfoot.asp) | Nhóm các nội dung footer trong một bảng |
| [<th>](https://www.w3schools.com/tags/tag_th.asp) | Định nghĩa một ô tiêu đề trong một bảng |
| [<thead>](https://www.w3schools.com/tags/tag_thead.asp) | Nhóm các nội dung tiêu đề trong một bảng |
| [<time>](https://www.w3schools.com/tags/tag_time.asp) | Định nghĩa ngày/giờ |
| [<title>](https://www.w3schools.com/tags/tag_title.asp) | Định nghĩa một tiêu đề cho tài liệu |
| [<tr>](https://www.w3schools.com/tags/tag_tr.asp) | Định nghĩa một hàng trong một bảng |
| [<track>](https://www.w3schools.com/tags/tag_track.asp) | Định nghĩa nội dung mô tả như chú thích, bình luận, hoặc các loại văn bản khác cho tập tin đa phương tiện (<video> và <audio>) |
| [<tt>](https://www.w3schools.com/tags/tag_tt.asp) | Không hỗ trợ trong HTML5. Sử dụng CSS để thay thế.  Xác định văn bản của máy |
| [<u>](https://www.w3schools.com/tags/tag_u.asp) | Sử dụng để tạo chữ gạch chân cho phần văn bản |
| [<ul>](https://www.w3schools.com/tags/tag_ul.asp) | Định nghĩa một danh sách k có thứ tự |
| [<var>](https://www.w3schools.com/tags/tag_var.asp) | Định nghĩa một biến |
| [<video>](https://www.w3schools.com/tags/tag_video.asp) | Định nghĩa một video hoặc movie |
| [<wbr>](https://www.w3schools.com/tags/tag_wbr.asp) | Định nghĩa một dấu ngắt dòng |

Các thuộc tính HTML :

|  |  |  |
| --- | --- | --- |
| Thuộc tính | Cho thẻ | Mô tả |
| [accept](https://www.w3schools.com/tags/att_accept.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp) | Chỉ định các loại tập tin mà máy chủ chấp nhận (chỉ dành cho thẻ input có type= “file”) |
| [accept-charset](https://www.w3schools.com/tags/att_accept-charset.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp) | Chỉ định kiểu mã ký tự được sử dụng cho form gửi dữ liệu đi. |
| [accesskey](https://www.w3schools.com/tags/att_accesskey.asp) | [Tất](https://www.w3schools.com/tags/ref_standardattributes.asp) cả các thẻ | Chỉ định một phím tắt đêt kích hoạt/focus vào một phần tử |
| [action](https://www.w3schools.com/tags/att_action.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp) | Chỉ định nơi để gửi dữ liệu trên form khi một form được kích nút gửi. |
| align | Không hỗ trợ trong HTML5 | Chỉ định việc căn chỉnh dữ liệu của các phần tử. thường sử dụng trong CSS. |
| [alt](https://www.w3schools.com/tags/att_alt.asp) | [<area>](https://www.w3schools.com/tags/tag_area.asp), [<img>](https://www.w3schools.com/tags/tag_img.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp) | Chỉ định văn bản thay thế cho các phần tử không hiển thị, ví dụ như phần tử img ta dùng alt để mô tả thông tin cho ảnh để các công cụ tìm kiếm dễ tìm thấy. |
| [async](https://www.w3schools.com/tags/att_async.asp) | [<script>](https://www.w3schools.com/tags/tag_script.asp) | Chỉ định kịch bản không đồng bộ (chỉ cho các script bên ngoài). |
| [autocomplete](https://www.w3schools.com/tags/att_autocomplete.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp) | Chỉ định một form hoặc một phần tử input có thiết lập thuộc tính autocomplete hay không, ví dụ như gõ vào ô textbox, nếu thiết lập autocomplete nó sẽ gợi ý các từ đã nhập trước đây. |
| [autofocus](https://www.w3schools.com/tags/att_autofocus.asp) | [<button>](https://www.w3schools.com/tags/tag_button.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<select>](https://www.w3schools.com/tags/tag_select.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Xác định các phần tử sẽ tự động nhận focus khi tải lại trang |
| [autoplay](https://www.w3schools.com/tags/att_autoplay.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Chỉ định âm thanh/video sẽ bắt đầu chạy khi nó đã sẵn sang. |
| bgcolor | Không hỗ trợ trong HTML5. | Xác định màu nền của một phần tử. Thường sử dụng trong CSS. |
| border | Không hỗ trợ trong HTML5. | Xác định độ rộng đường giới của một phần tử. Thường dùng trong CSS. |
| [charset](https://www.w3schools.com/tags/att_charset.asp) | [<meta>](https://www.w3schools.com/tags/tag_meta.asp), [<script>](https://www.w3schools.com/tags/tag_script.asp) | Xác định bộ mã ký tự cho văn bản HTML |
| [checked](https://www.w3schools.com/tags/att_checked.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp) | Chỉ định một phần tử <input> là được chọn khi tải lại trang (đối với type = “checkbox” hoặc type = “radio”). |
| [cite](https://www.w3schools.com/tags/att_cite.asp) | <blockquote>, [<del>](https://www.w3schools.com/tags/tag_del.asp), [<ins>](https://www.w3schools.com/tags/tag_ins.asp), <q> | Chỉ định một đường dẫn URL giải thích việc trích dẫn/xóa/chèn văn bản vào. |
| [class](https://www.w3schools.com/tags/att_class.asp) | Tất cả các thẻ | Chỉ định một hoặc nhiều tên class cho một phần tử (đề cập đến 1 class trong 1 stylesheet). |
| color | Không hỗ trợ trong HTML5 | Xác định màu chữ của một phần tử. Thường dùng trong CSS. |
| [cols](https://www.w3schools.com/tags/att_cols.asp) | [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Xác định chiều rộng cột cho phần tử text area. |
| [colspan](https://www.w3schools.com/tags/att_colspan.asp) | [<td>](https://www.w3schools.com/tags/tag_td.asp), [<th>](https://www.w3schools.com/tags/tag_th.asp) | Xác định só cột của các ô trong bảng sẽ được gộp lại. |
| [content](https://www.w3schools.com/tags/att_content.asp) | [<meta>](https://www.w3schools.com/tags/tag_meta.asp) | Cung cấp giá trị liên quan với http-equiv hoặc thuộc tính name. |
| [contenteditable](https://www.w3schools.com/tags/att_contenteditable.asp) | Tất cả các thẻ | Xác định nội dung của một phần tử có thể biên tập được hay không. |
| [controls](https://www.w3schools.com/tags/att_controls.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Xác định trình điều khiển âm thanh/video sẽ được hiển thị (ví dụ như một nút play/pause…). |
| [coords](https://www.w3schools.com/tags/att_coords.asp) | [<area>](https://www.w3schools.com/tags/tag_area.asp) | Xác định tọa độ của một vùng. |
| [data](https://www.w3schools.com/tags/att_data.asp) | [<object>](https://www.w3schools.com/tags/tag_object.asp) | Chỉ định URL của một tài nguyên được sử dụng bởi một đối tượng. |
| [data-\*](https://www.w3schools.com/tags/att_data-.asp) | Tất cả các thẻ | Được sử dụng để lưu trữ dữ liệu tùy biến riêng cho các trang hoặc ứng dụng. |
| [datetime](https://www.w3schools.com/tags/att_datetime.asp) | [<del>](https://www.w3schools.com/tags/tag_del.asp), [<ins>](https://www.w3schools.com/tags/tag_ins.asp), [<time>](https://www.w3schools.com/tags/tag_time.asp) | Xác định thời gian. |
| [default](https://www.w3schools.com/tags/att_default.asp) | [<track>](https://www.w3schools.com/tags/tag_track.asp) | Là thuộc tính kiểu Boolean, xác định track này là được kích hoạt nếu người dùng không chỉ rõ một track khác thích hợp hơn. |
| [defer](https://www.w3schools.com/tags/att_defer.asp) | [<script>](https://www.w3schools.com/tags/tag_script.asp) | Chỉ định rằng script đã được thực thi khi trang đã hoàn thành phân tích (chỉ dành cho kịch bản bên ngoài). |
| [dir](https://www.w3schools.com/tags/att_dir.asp) | Tất cả các thẻ | Chỉ định hướng của văn bản cho nội dung trong một phần tử ví dụ trái sang phải, phải qua trái… |
| [dirname](https://www.w3schools.com/tags/att_dirname.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Xác định hướng của văn bản khi được submit. |
| [disabled](https://www.w3schools.com/tags/att_disabled.asp) | [<button>](https://www.w3schools.com/tags/tag_button.asp), [<fieldset>](https://www.w3schools.com/tags/tag_fieldset.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), <optgroup>, [<option>](https://www.w3schools.com/tags/tag_option.asp), [<select>](https://www.w3schools.com/tags/tag_select.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Chỉ định một hoặc một nhóm các phần tử sẽ bị vô hiệu hóa. |
| [download](https://www.w3schools.com/tags/att_download.asp) | [<a>](https://www.w3schools.com/tags/tag_a.asp), [<area>](https://www.w3schools.com/tags/tag_area.asp) | Chỉ định đích được tải tới khi người dùng nhấn vào một liên kết. |
| [draggable](https://www.w3schools.com/tags/att_draggable.asp) | [Tất](https://www.w3schools.com/tags/ref_standardattributes.asp) cả các thẻ | Chỉ định một phần tử có thể kéo thả hay không. |
| [dropzone](https://www.w3schools.com/tags/att_dropzone.asp) | [Tất](https://www.w3schools.com/tags/ref_standardattributes.asp) cả các thẻ | Chỉ định dữ liệu được kéo là sao chép, di chuyển, hoặc liên kết khi thả ra (dùng cho sự kiện kéo thả). |
| [enctype](https://www.w3schools.com/tags/att_enctype.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp) | Specifies how the form-data should be encoded when submitting it to the server (only for method="post") |
| [for](https://www.w3schools.com/tags/att_for.asp) | <label>, [<output>](https://www.w3schools.com/tags/tag_output.asp) | Chỉ định các phần tử của form của một nhãn hoặc một phép tính có rang buộc tới. |
| [form](https://www.w3schools.com/tags/att_form.asp) | [<button>](https://www.w3schools.com/tags/tag_button.asp), [<fieldset>](https://www.w3schools.com/tags/tag_fieldset.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), <label>, [<meter>](https://www.w3schools.com/tags/tag_meter.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<output>](https://www.w3schools.com/tags/tag_output.asp), [<select>](https://www.w3schools.com/tags/tag_select.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Chỉ định các phần tử được chứa trong một form có tên nào đó. |
| [formaction](https://www.w3schools.com/tags/att_formaction.asp) | [<button>](https://www.w3schools.com/tags/tag_button.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp) | Chỉ định nơi để gửi dữ liệu của form khi một form được submit. Chỉ dành cho type= “submit”. |
| [headers](https://www.w3schools.com/tags/att_headers.asp) | [<td>](https://www.w3schools.com/tags/tag_td.asp), [<th>](https://www.w3schools.com/tags/tag_th.asp) | Chỉ định một hoặc nhiều ô ở phần tiêu đề của bảng có liên quan đến với các ô dữ liệu trong bảng. |
| [height](https://www.w3schools.com/tags/att_height.asp) | <canvas>, [<embed>](https://www.w3schools.com/tags/tag_embed.asp), [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp), [<img>](https://www.w3schools.com/tags/tag_img.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Chỉ định chiều cao của phần tử. |
| [hidden](https://www.w3schools.com/tags/att_hidden.asp) | [Tất](https://www.w3schools.com/tags/ref_standardattributes.asp) cả các thẻ | Là thuộc tính Boolean, chỉ định một phần tử không được hiển thị. |
| [high](https://www.w3schools.com/tags/att_high.asp) | [<meter>](https://www.w3schools.com/tags/tag_meter.asp) | Xác định giá trị một khoảng nào đó và được xem như là một giá trị độ cao. |
| [href](https://www.w3schools.com/tags/att_href.asp) | [<a>](https://www.w3schools.com/tags/tag_a.asp), [<area>](https://www.w3schools.com/tags/tag_area.asp), [<base>](https://www.w3schools.com/tags/tag_base.asp), [<link>](https://www.w3schools.com/tags/tag_link.asp) | Chỉ định URL của trang liên kết đi tới. |
| [hreflang](https://www.w3schools.com/tags/att_hreflang.asp) | [<a>](https://www.w3schools.com/tags/tag_a.asp), [<area>](https://www.w3schools.com/tags/tag_area.asp), [<link>](https://www.w3schools.com/tags/tag_link.asp) | Chỉ định ngôn ngữ của tài liệu được liên kết. |
| [http-equiv](https://www.w3schools.com/tags/att_http-equiv.asp) | [<meta>](https://www.w3schools.com/tags/tag_meta.asp) | Cung cấp phần tiêu đề HTTP cho các thông tin/giá trị của thuộc tính nội dung. |
| [id](https://www.w3schools.com/tags/att_id.asp) | [Tất](https://www.w3schools.com/tags/ref_standardattributes.asp) cả các thẻ | Chỉ định một id duy nhất cho một phần tử. |
| [ismap](https://www.w3schools.com/tags/att_ismap.asp) | [<img>](https://www.w3schools.com/tags/tag_img.asp) | Chỉ định một hình ảnh như một server-side image-map. |
| [kind](https://www.w3schools.com/tags/att_kind.asp) | [<track>](https://www.w3schools.com/tags/tag_track.asp) | Chỉ định loại text cho phần tử track. |
| [label](https://www.w3schools.com/tags/att_label.asp) | [<track>](https://www.w3schools.com/tags/tag_track.asp), [<option>](https://www.w3schools.com/tags/tag_option.asp), [<optgroup>](https://www.w3schools.com/tags/tag_optgroup.asp) | Chỉ định tiêu đề của text cho phần tử track. |
| [lang](https://www.w3schools.com/tags/att_lang.asp) | [Tất](https://www.w3schools.com/tags/ref_standardattributes.asp) cả các thẻ | Chỉ định ngôn ngữ cho phần nội dung của phần tử. |
| [list](https://www.w3schools.com/tags/att_list.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp) | Đề cập đến phần tử <datalisr> mà chứa các giá trị tùy chọn định nghĩa trước cho một phàn tử input. |
| [loop](https://www.w3schools.com/tags/att_loop.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Chỉ định các âm thanh/video sẽ lặp lại một lần nữa mỗi khi nó chạy hết. |
| [low](https://www.w3schools.com/tags/att_low.asp) | [<meter>](https://www.w3schools.com/tags/tag_meter.asp) | Xác định khoảng giá trị thấp cho phần tử meter |
| [max](https://www.w3schools.com/tags/att_max.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<meter>](https://www.w3schools.com/tags/tag_meter.asp), [<progress>](https://www.w3schools.com/tags/tag_progress.asp) | Xác định giá trị lớn nhất cho phần tử. |
| [maxlength](https://www.w3schools.com/tags/att_maxlength.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Xác định số lượng ký tự tối đa cho phép trong một phần tử. |
| [media](https://www.w3schools.com/tags/att_media.asp) | [<a>](https://www.w3schools.com/tags/tag_a.asp), [<area>](https://www.w3schools.com/tags/tag_area.asp), [<link>](https://www.w3schools.com/tags/tag_link.asp), [<source>](https://www.w3schools.com/tags/tag_source.asp), [<style>](https://www.w3schools.com/tags/tag_style.asp) | Chỉ định những gì truyền thông/thiết bị các tài liệu liên quan được tối ưu hóa cho. |
| [method](https://www.w3schools.com/tags/att_method.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp) | Chỉ định các phương thức HTTP để sử dụng khi gửi form-data. |
| [min](https://www.w3schools.com/tags/att_min.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<meter>](https://www.w3schools.com/tags/tag_meter.asp) | Chỉ định một giá trị tối thiểu. |
| [multiple](https://www.w3schools.com/tags/att_multiple.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<select>](https://www.w3schools.com/tags/tag_select.asp) | Chỉ định rằng người dùng có thể nhập vào nhiều hơn một giá trị. |
| [muted](https://www.w3schools.com/tags/att_muted.asp) | [<video>](https://www.w3schools.com/tags/tag_video.asp), [<audio>](https://www.w3schools.com/tags/tag_audio.asp) | Chỉ định rằng đầu ra âm thanh của video nên được tắt. |
| [name](https://www.w3schools.com/tags/att_name.asp) | [<button>](https://www.w3schools.com/tags/tag_button.asp), [<fieldset>](https://www.w3schools.com/tags/tag_fieldset.asp), [<form>](https://www.w3schools.com/tags/tag_form.asp), [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<map>](https://www.w3schools.com/tags/tag_map.asp), [<meta>](https://www.w3schools.com/tags/tag_meta.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<output>](https://www.w3schools.com/tags/tag_output.asp), [<param>](https://www.w3schools.com/tags/tag_param.asp), [<select>](https://www.w3schools.com/tags/tag_select.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Chỉ định tên của phần tử. |
| [novalidate](https://www.w3schools.com/tags/att_novalidate.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp) | Chỉ định form không cần kiểm tra tính hợp lệ (validate) trước khi được submit. |
| [onabort](https://www.w3schools.com/tags/att_onabort.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<embed>](https://www.w3schools.com/tags/tag_embed.asp), [<img>](https://www.w3schools.com/tags/tag_img.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script này được chạy khi hủy bỏ. |
| [onafterprint](https://www.w3schools.com/tags/att_onafterprint.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script này được chạy sau khi các tài liệu được in. |
| [onbeforeprint](https://www.w3schools.com/tags/att_onbeforeprint.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script này được chạy trước khi các tài liệu được in. |
| [onbeforeunload](https://www.w3schools.com/tags/att_onbeforeunload.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script này được chạy khi tài liệu sắp được gỡ bỏ. |
| [onblur](https://www.w3schools.com/tags/att_onblur.asp) | Có ở tất cả các phần tử | Script này được chạy khi phần tử mất focus. |
| [oncanplay](https://www.w3schools.com/tags/att_oncanplay.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<embed>](https://www.w3schools.com/tags/tag_embed.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script này được chạy khi một tập tin đã sẵn sang để bắt đầu (file đã nạp đủ trong bộ để chạy) |
| [oncanplaythrough](https://www.w3schools.com/tags/att_oncanplaythrough.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script này được chạy khi một tập tin có thể được chạy theo tất cả các cách khác nhau đến kết thúc mà không dừng lại nạp vào bộ đệm. |
| [onchange](https://www.w3schools.com/tags/att_onchange.asp) | Có ở tất cả các phần tử | Script này được chạy khi giá trị của phần tử bị thay đổi. |
| [onclick](https://www.w3schools.com/tags/att_onclick.asp) | Có ở tất cả các phần tử. | Script này được chạy khi giá trị của phần tử được kích vào. |
| [oncontextmenu](https://www.w3schools.com/tags/att_oncontextmenu.asp) | Có ở tất cả các phần tử. | Script này được chạy khi một trình đơn ngữ cảnh được kích hoạt. |
| [oncopy](https://www.w3schools.com/tags/att_oncopy.asp) | Có ở tất cả các phần tử. | Script này được chạy khi nội dung của phần tử đang được sao chép. |
| [oncuechange](https://www.w3schools.com/tags/att_oncuechange.asp) | [<track>](https://www.w3schools.com/tags/tag_track.asp) | Script này được chạy khi có dấu hiệu thay đổi trong phần tử <track> |
| [oncut](https://www.w3schools.com/tags/att_oncut.asp) | Có ở tất cả các phần tử. | Script này được chạy khi nội dung của phần tử đang được cắt giảm. |
| [ondblclick](https://www.w3schools.com/tags/att_ondblclick.asp) | Có ở tất cả các phần tử. | Script này được chạy khi phần tử này được nhấn đúp chuột. |
| [ondrag](https://www.w3schools.com/tags/att_ondrag.asp) | Có ở tất cả các phần tử. | Script này được chạy ở phần cuối của một hoạt động kéo. |
| [ondragend](https://www.w3schools.com/tags/att_ondragend.asp) | Có ở tất cả các phần tử. | Script này được chạy khi kết thúc một hoạt động kéo. |
| [ondragenter](https://www.w3schools.com/tags/att_ondragenter.asp) | Có ở tất cả các phần tử. | Script được chạy khi một phần tử đã được kéo đến một mục tiêu thả hợp lệ. |
| [ondragleave](https://www.w3schools.com/tags/att_ondragleave.asp) | Có ở tất cả các phần tử. | Script này được chạy khi một phần tử rời khỏi một mục tiêu thả hợp lệ. |
| [ondragover](https://www.w3schools.com/tags/att_ondragover.asp) | Có ở tất cả các phần tử. | Script này được chạy khi một phần tử đang được kéo qua một mục tiêu thả hợp lệ. |
| [ondragstart](https://www.w3schools.com/tags/att_ondragstart.asp) | Có ở tất cả các phần tử. | Script này được chạy khi bắt đầu một hoạt động kéo. |
| [ondrop](https://www.w3schools.com/tags/att_ondrop.asp) | Có ở tất cả các phần tử. | Script được chạy khi yếu tố kéo đang được giảm. |
| [ondurationchange](https://www.w3schools.com/tags/att_ondurationchange.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script này được chạy khi thay đổi chiều dài của những phương tiện truyền thông. |
| [onemptied](https://www.w3schools.com/tags/att_onemptied.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script này được chạy khi một cái gì xấu xảy ra và tập tin là đột nhiên không có sẵn (như bất ngờ ngắt kết nối). |
| [onended](https://www.w3schools.com/tags/att_onended.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi các phương tiện truyền thông đã được kết thúc ( một sự kiện hữu ích cho những thông điệp như “cảm ơn đã nghe”). |
| [onerror](https://www.w3schools.com/tags/att_onerror.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<body>](https://www.w3schools.com/tags/tag_body.asp), [<embed>](https://www.w3schools.com/tags/tag_embed.asp), [<img>](https://www.w3schools.com/tags/tag_img.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<script>](https://www.w3schools.com/tags/tag_script.asp), [<style>](https://www.w3schools.com/tags/tag_style.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi một lỗi xảy ra. |
| [onfocus](https://www.w3schools.com/tags/att_onfocus.asp) | Có ở tất cả các phần tử. | Script được chạy khi các yếu tố được tập trung. |
| [onhashchange](https://www.w3schools.com/tags/att_onhashchange.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi đã có những thay đổi trong phần neo của một URL. |
| [oninput](https://www.w3schools.com/tags/att_oninput.asp) | Có ở tất cả các phần tử. | Script được chạy khi các yếu tố đầu vào được sử dụng. |
| [oninvalid](https://www.w3schools.com/tags/att_oninvalid.asp) | Có ở tất cả các phần tử. | Script được chạy khi các phần tử không hợp lệ. |
| [onkeydown](https://www.w3schools.com/tags/att_onkeydown.asp) | Có ở tất cả các phần tử. | Script được chạy khi người dùng nhấn một phím bất kỳ. |
| [onkeypress](https://www.w3schools.com/tags/att_onkeypress.asp) | Có ở tất cả các phần tử. | Script được chạy khi người dùng nhấn một phím bất kỳ. |
| [onkeyup](https://www.w3schools.com/tags/att_onkeyup.asp) | Có ở tất cả các phần tử. | Script được chạy khi người dùng nhả một phím. |
| [onload](https://www.w3schools.com/tags/att_onload.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp), [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp), [<img>](https://www.w3schools.com/tags/tag_img.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<link>](https://www.w3schools.com/tags/tag_link.asp), [<script>](https://www.w3schools.com/tags/tag_script.asp), [<style>](https://www.w3schools.com/tags/tag_style.asp) | Script được chạy khi các phần tử được tải xong. |
| [onloadeddata](https://www.w3schools.com/tags/att_onloadeddata.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy ki dữ liệu truyền thông được tải. |
| [onloadedmetadata](https://www.w3schools.com/tags/att_onloadedmetadata.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi siêu dữ liệu (như kích thước và thời gian) được nạp. |
| [onloadstart](https://www.w3schools.com/tags/att_onloadstart.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy như các tập tin để bắt đầu tải trước khi bất cứ điều gì là thực sự nạp. |
| [onmousedown](https://www.w3schools.com/tags/att_onmousedown.asp) | Có ở tất cả các phần tử. | Script được chạy khi một nút chuột được nhấn xuống trên một phần tử. |
| [onmousemove](https://www.w3schools.com/tags/att_onmousemove.asp) | Có ở tất cả các phần tử. | Script được chạy khi con trỏ chuột di chuyển trên một phần tử. |
| [onmouseout](https://www.w3schools.com/tags/att_onmouseout.asp) | Có ở tất cả các phần tử. | Script được chạy khi một con trỏ chuột di chuyển khỏi các phần tử. |
| [onmouseover](https://www.w3schools.com/tags/att_onmouseover.asp) | Có ở tất cả các phần tử. | Script được chạy khi một con trỏ chuột di chuyển trên một phần tử. |
| [onmouseup](https://www.w3schools.com/tags/att_onmouseup.asp) | Có ở tất cả các phần tử. | Script được chạy khi một nút chuột được nhấn trên một phần tử. |
| [onmousewheel](https://www.w3schools.com/tags/att_onmousewheel.asp) | Có ở tất cả các phần tử. | Script được chạy khi một bánh xe chuột đang được cuộn trên một phần tử. |
| [onoffline](https://www.w3schools.com/tags/att_onoffline.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi trình duyệt bắt đầu làm việc chế độ offline |
| [ononline](https://www.w3schools.com/tags/att_ononline.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi trình duyệt bắt đầu làm việc chế độ online. |
| onpagehide | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi người dùng điều hướng ra khỏi một trang. |
| [onpageshow](https://www.w3schools.com/tags/att_onpageshow.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi người dùng điều hướng đến một trang. |
| [onpaste](https://www.w3schools.com/tags/att_onpaste.asp) | Có ở tất cả các phần tử. | Script được chạy khi người dùng dán một số nội dung vào một phần tử. |
| [onpause](https://www.w3schools.com/tags/att_onpause.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi tập tin đa phương tiện bị tạm dừng hoặc bởi người sử dụng hoặc lập trình. |
| [onplay](https://www.w3schools.com/tags/att_onplay.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi tập tin đa phương tiện đã sẵn sàng để chạy. |
| [onplaying](https://www.w3schools.com/tags/att_onplaying.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi tập tin đa phương tiện thực sự đã bắt đầu chạy. |
| onpopstate | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi lịch sử của cửa sổ thay đổi. |
| [onprogress](https://www.w3schools.com/tags/att_onprogress.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi trình duyệt đang trong quá trình xử lý nhận dữ liệu đa phương tiện. |
| [onratechange](https://www.w3schools.com/tags/att_onratechange.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy mỗi khi thay đổi tốc độ phát lại (giống như khi người dùng chuyển sang một chế độ chuyển động chậm hoặc chế độ chuyển tiếp nhanh). |
| [onreset](https://www.w3schools.com/tags/att_onreset.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp) | Script được chạy khi nút reset trên form được kích chuột. |
| [onresize](https://www.w3schools.com/tags/att_onresize.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi cửa sổ trình duyệt bị thay đổi kích thước. |
| [onscroll](https://www.w3schools.com/tags/att_onscroll.asp) | Có ở tất cả các phần tử. | Script được chạy khi thanh cuộn của phần tử được cuộn. |
| [onsearch](https://www.w3schools.com/tags/att_onsearch.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp) | Script được chạy khi người dùng viết một cái gì đó trong trường search (<input="search">) |
| [onseeked](https://www.w3schools.com/tags/att_onseeked.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi các thuộc tính seeking được thiết lập là false để chỉ tìm kiếm đã kết thúc. |
| [onseeking](https://www.w3schools.com/tags/att_onseeking.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi các thuộc tính seeking được thiết lập là true để chỉ tìm kiếm được kích hoạt. |
| [onselect](https://www.w3schools.com/tags/att_onselect.asp) | Có ở tất cả các phần tử. | Script được chạy khi các phần tử được chọn. |
| [onstalled](https://www.w3schools.com/tags/att_onstalled.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi trình duyệt là không thể lấy dữ liệu đa phương tiện vì bất kỳ lý do gì. |
| onstorage | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi vùng lưu trữ web bị cập nhật. |
| [onsubmit](https://www.w3schools.com/tags/att_onsubmit.asp) | [<form>](https://www.w3schools.com/tags/tag_form.asp) | Script được chạy khi form được submit. |
| [onsuspend](https://www.w3schools.com/tags/att_onsuspend.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi việc nạp dữ liệu đa phương tiện bị dừng lại trước khi nó hoàn thành việc tải về với bất kỳ lý do gì |
| [ontimeupdate](https://www.w3schools.com/tags/att_ontimeupdate.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi các vị trí đang chạy bị thay đổi (như khi người dùng chuyển nhanh đến một điểm khác nhau của tệp tin đa phương tiện) |
| [ontoggle](https://www.w3schools.com/tags/att_ontoggle.asp) | [<details>](https://www.w3schools.com/tags/tag_details.asp) | Script được chạy khi người dùng mở hoặc đóng phần tử <detail>. |
| [onunload](https://www.w3schools.com/tags/att_onunload.asp) | [<body>](https://www.w3schools.com/tags/tag_body.asp) | Script được chạy khi một trang đã được giải phóng (hoặc cửa sổ trình duyệt đã bị đóng). |
| [onvolumechange](https://www.w3schools.com/tags/att_onvolumechange.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy mỗi khi âm lượng được thay đổi. |
| [onwaiting](https://www.w3schools.com/tags/att_onwaiting.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Script được chạy khi tập tin đa phương tiện đang tạm dừng nhưng dự kiến sẽ tiếp tục (giống như tập tin đa phương tiện dừng để đợi bộ đệm nhiều dữ liệu hơn). |
| [onwheel](https://www.w3schools.com/tags/att_onwheel.asp) | Có ở tất cả các phần tử. | Script được chạy khi con lăn ở giữa chuột cuộn lê hoặc xuống trên một phần tử. |
| [open](https://www.w3schools.com/tags/att_open.asp) | [<details>](https://www.w3schools.com/tags/tag_details.asp) | Chỉ định phần thông tin chi tiết sẽ được mở (hiển thị) cho người dùng. |
| [optimum](https://www.w3schools.com/tags/att_optimum.asp) | [<meter>](https://www.w3schools.com/tags/tag_meter.asp) | Chỉ định giá trị tối ưu cho phần tử meter. |
| [pattern](https://www.w3schools.com/tags/att_pattern.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp) | Chỉ định một biểu thức chính quy mà giá trị của phần tử <input> được kiểm tra. |
| [placeholder](https://www.w3schools.com/tags/att_placeholder.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Chỉ định một gợi ý ngắn mô tả chức năng cho phần tử. |
| [poster](https://www.w3schools.com/tags/att_poster.asp) | [<video>](https://www.w3schools.com/tags/tag_video.asp) | Chỉ định một hình ảnh được hiển thị trong khi video được tải về, hoặc cho đến khi người dùng chạm vào nút play. |
| [preload](https://www.w3schools.com/tags/att_preload.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Chỉ định các file âm thanh/video có được tải khi trang được tải hay không. |
| [readonly](https://www.w3schools.com/tags/att_readonly.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Chỉ định rằng các phần tử chỉ được đọc. |
| [rel](https://www.w3schools.com/tags/att_rel.asp) | [<a>](https://www.w3schools.com/tags/tag_a.asp), [<area>](https://www.w3schools.com/tags/tag_area.asp), [<link>](https://www.w3schools.com/tags/tag_link.asp) | Xác định mối quan hệ giữa các tài liệu hiện hành và các tài liệu được liên kết. |
| [required](https://www.w3schools.com/tags/att_required.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<select>](https://www.w3schools.com/tags/tag_select.asp), [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Chỉ định rằng phần tử này phải được điền đầ đủ trước khi form được submit. |
| [reversed](https://www.w3schools.com/tags/att_reversed.asp) | [<ol>](https://www.w3schools.com/tags/tag_ol.asp) | Chỉ định rằng thứ tự danh sách nên giảm dần (9,8,7…). |
| [rows](https://www.w3schools.com/tags/att_rows.asp) | [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Xác định số lượng dòng có trong một TextArea. |
| [rowspan](https://www.w3schools.com/tags/att_rowspan.asp) | [<td>](https://www.w3schools.com/tags/tag_td.asp), [<th>](https://www.w3schools.com/tags/tag_th.asp) | Xác định số lượng ô trong một hàng của bảng được gộp. |
| [sandbox](https://www.w3schools.com/tags/att_sandbox.asp) | [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp) | Cho phép thêm các hạn chế đối với các nội dung trong một <iframe> |
| [scope](https://www.w3schools.com/tags/att_scope.asp) | [<th>](https://www.w3schools.com/tags/tag_th.asp) | Chỉ định tiêu đề của một ô là một tiêu đề cho một cột, một hàng, hay một nhóm của các cột hoặc các hàng. |
| [selected](https://www.w3schools.com/tags/att_selected.asp) | [<option>](https://www.w3schools.com/tags/tag_option.asp) | Chỉ định một giá trị được chọn mặc định khi tải trang. |
| [shape](https://www.w3schools.com/tags/att_shape.asp) | [<area>](https://www.w3schools.com/tags/tag_area.asp) | Chỉ định hình dạng của phần tử area. |
| [size](https://www.w3schools.com/tags/att_size.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp), [<select>](https://www.w3schools.com/tags/tag_select.asp) | Xác định chiều rộng hay số lượng ký tự (cho thẻ <input>) hoặc số lượng hiển thị ccas giá trị tùy chọn (cho thẻ <select>). |
| [sizes](https://www.w3schools.com/tags/att_sizes.asp) | [<img>](https://www.w3schools.com/tags/tag_img.asp), [<link>](https://www.w3schools.com/tags/tag_link.asp), [<source>](https://www.w3schools.com/tags/tag_source.asp) | Chỉ định kích thước của tài nguyên được liên kết tới. |
| [span](https://www.w3schools.com/tags/att_span.asp) | [<col>](https://www.w3schools.com/tags/tag_col.asp), [<colgroup>](https://www.w3schools.com/tags/tag_colgroup.asp) | Xác định số lượng cột được gộp lại thành một. |
| [spellcheck](https://www.w3schools.com/tags/att_spellcheck.asp) | Tất cả các thẻ | Chỉ định các phần tử có kiểm tra lỗi chính tả và ngữ pháp hay không. |
| [src](https://www.w3schools.com/tags/att_src.asp) | [<audio>](https://www.w3schools.com/tags/tag_audio.asp), [<embed>](https://www.w3schools.com/tags/tag_embed.asp), [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp), [<img>](https://www.w3schools.com/tags/tag_img.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<script>](https://www.w3schools.com/tags/tag_script.asp), [<source>](https://www.w3schools.com/tags/tag_source.asp), [<track>](https://www.w3schools.com/tags/tag_track.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Chỉ định URL của tập tin đa phương tiện. |
| [srcdoc](https://www.w3schools.com/tags/att_srcdoc.asp) | [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp) | Chỉ định các nội dung HTML của trang hiển thị trong thẻ <iframe>. |
| [srclang](https://www.w3schools.com/tags/att_srclang.asp) | [<track>](https://www.w3schools.com/tags/tag_track.asp) | Chỉ định ngôn ngữ dữ liệu văn bản của track (được yêu cầu type = “subtitle”). |
| [srcset](https://www.w3schools.com/tags/att_source_srcset.asp) | [<img>](https://www.w3schools.com/tags/tag_img.asp), [<source>](https://www.w3schools.com/tags/tag_source.asp) | Chỉ định URL của hình ảnh để sử dụng trong các tình huống khác nhau. |
| [start](https://www.w3schools.com/tags/att_start.asp) | [<ol>](https://www.w3schools.com/tags/tag_ol.asp) | Chỉ định giá trị bắt đầu của danh sách đầu mục là số. |
| [step](https://www.w3schools.com/tags/att_step.asp) | [<input>](https://www.w3schools.com/tags/tag_input.asp) | Chỉ định bước nhảy với dữ liệu số cho trường input. |
| [style](https://www.w3schools.com/tags/att_style.asp) | Tất cả các thẻ | Chỉ định style CSS inline ho một phần tử |
| [tabindex](https://www.w3schools.com/tags/att_tabindex.asp) | Tất cả các thẻ | Chỉ định thứ tự khi ấn phím tab cho một phần tử. |
| [target](https://www.w3schools.com/tags/att_target.asp) | [<a>](https://www.w3schools.com/tags/tag_a.asp), [<area>](https://www.w3schools.com/tags/tag_area.asp), [<base>](https://www.w3schools.com/tags/tag_base.asp), [<form>](https://www.w3schools.com/tags/tag_form.asp) | Xác định đích khi mở các tài liệu, liên qaun hoặc nơi các form được submit. |
| [title](https://www.w3schools.com/tags/att_title.asp) | Tất cả các thẻ | Thêm thông tin về một phần tử. |
| [translate](https://www.w3schools.com/tags/att_translate.asp) | Tất cả các thẻ | Chỉ định nội dung của một phần tử phải được dịch hay không. |
| [type](https://www.w3schools.com/tags/att_type.asp) | [<a>](https://www.w3schools.com/tags/tag_a.asp), [<button>](https://www.w3schools.com/tags/tag_button.asp), [<embed>](https://www.w3schools.com/tags/tag_embed.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<link>](https://www.w3schools.com/tags/tag_link.asp), [<menu>](https://www.w3schools.com/tags/tag_menu.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<script>](https://www.w3schools.com/tags/tag_script.asp), [<source>](https://www.w3schools.com/tags/tag_source.asp), [<style>](https://www.w3schools.com/tags/tag_style.asp) | Chỉ định kiểu của phần tử. |
| [usemap](https://www.w3schools.com/tags/att_usemap.asp) | [<img>](https://www.w3schools.com/tags/tag_img.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp) | Chỉ định việc ánh xạ một hình ảnh owrphias client-side image-map. |
| [value](https://www.w3schools.com/tags/att_value.asp) | [<button>](https://www.w3schools.com/tags/tag_button.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<li>](https://www.w3schools.com/tags/tag_li.asp), [<option>](https://www.w3schools.com/tags/tag_option.asp), [<meter>](https://www.w3schools.com/tags/tag_meter.asp), [<progress>](https://www.w3schools.com/tags/tag_progress.asp), [<param>](https://www.w3schools.com/tags/tag_param.asp) | Xác định giá trị của các phần tử. |
| [width](https://www.w3schools.com/tags/att_width.asp) | <canvas>, [<embed>](https://www.w3schools.com/tags/tag_embed.asp), [<iframe>](https://www.w3schools.com/tags/tag_iframe.asp), [<img>](https://www.w3schools.com/tags/tag_img.asp), [<input>](https://www.w3schools.com/tags/tag_input.asp), [<object>](https://www.w3schools.com/tags/tag_object.asp), [<video>](https://www.w3schools.com/tags/tag_video.asp) | Xác định chiều rộng của phần tử. |
| [wrap](https://www.w3schools.com/tags/att_wrap.asp) | [<textarea>](https://www.w3schools.com/tags/tag_textarea.asp) | Chỉ định cách hiển thị văn bản trong textarea. |

## JavaScript

### Căn bản về JavaScipt

#### Tổng quan về JavaScript

JavaScript có thể thay đổi nội dung của HTML một cách dễ dàng, Một trong những hàm JavaScript thông dụng là getElementById(), khi sử dụng hàm này JavaScript sẽ tìm các thành phần tương ứng, từ đó có thể lấy nội dung cũng như thay đổi nội dung của các thành phần này. Ví dụ với đoạn mã thẻ p có id là “demo”:

<p id="demo">JavaScript can change HTML content.</p>

Có thể sử dụng mã JavaScript sau để thay đổi nội dung của thẻ này.

document.getElementById("demo").innerHTML = "Hello JavaScript";

Giống HTML, có thể sử dụng cả dấu “ hoặc dấu ‘ thay thế cho nhau, ví dụ với câu lệnh trên, cũng có thể viết:

document.getElementById('demo').innerHTML = 'Hello JavaScript';

JavaScript co thể thay đổi thuộc tính của các thẻ HTML, ví dụ với đoạn mã dưới đây có thể được sử dụng để thay thế thuộc tính src của thành phần có id là myImage thành pic\_bulboff.gif

document.getElementById('myImage').src='pic\_bulboff.gif';

JavaScript cũng có thể thay đổi HTML Style (CSS) cũng tương tự như thay đổi thuộc tính của các thẻ HTML, ví dụ:

document.getElementById('demo').style.fontSize = '35px';

JavaScript có thể ẩn thành phần HTML ví dụ:

document.getElementById("demo").style.display = "none";

Hoặc hiện các thành phần HTML:

document.getElementById("demo").style.display = "block";

#### JS Where To

Các đoạn mã JavaScript sẽ được lập trình trong cặp thẻ <script> và </script>

<script>  
document.getElementById("demo").innerHTML = "My First JavaScript";  
</script>

Các thẻ script này có thể đặt ở bất kỳ vị trí nào trong file HTML, ngoài ra các đoạn mã JavaScript có thể để riêng ra file trong một file riêng. Khi đó các đoạn mã sẽ được lưu trong các file có phần mở rộng là JavaScript. Khi sử dụng file JavaScript này cần tham chiếu đến thông qua thuộc tính src của thẻ script, ví dụ:

<script src="myScript.js"></script>

Sử dụng các file JavaScript có nhiều lợi thế, nó giúp tách mã JavaScript và mã HTML. Điều này khiến các đoạn mã dễ đọc và bảo trì. Ngoài ra file JavaScript thường được lưu trong bộ đệm khiến việc tải trang được nhanh hơn. Ngoài ra các file script này có thể được truyền như một file được đặt trên cùng với file HTML hoặc truyền thông qua đường dẫn url:

<script src="https://www.w3schools.com/js/myScript1.js"></script>

#### JS Output

JavaScript có thể được sử dụng dữ liệu ra trang theo nhiều cách khác nhau, ví dụ để viết vào các thành phần HTML bằng cách sử dụng innerHTML, hay sử dụng hàm document.write(), hiển thị các thông báo (alert), hoặc ghi dữ liệu ra console.

Khi sử dụng innerHTML, người lập trình JavaScript thông thường sẽ sử dụng phương thức document.getElementById(id) để lấy thành phần HTML trước sau đó thay đổi nội dung thông qua innerHTML, như ví dụ sau:

<html>  
<body>  
<h1>My First Web Page</h1>  
<p>My First Paragraph</p>  
<p id="demo"></p>  
<script>  
document.getElementById("demo").innerHTML = 5 + 6;  
</script>  
</body>  
</html>

Hàm document.write() phù hợp cho mục đích test. Cần lưu ý: nếu sử dụng document.write sau khi trang HTML được tải, sẽ xoá toàn bộ các nội dung đang có trên file HTML.

<html>  
<body>  
<h1>My First Web Page</h1>  
<p>My first paragraph.</p>  
<script>  
document.write(5 + 6);  
</script>  
</body>  
</html>

Sử dụng alert() để hiện thị một hộp thoại

<html>  
<body>  
<h1>My First Web Page</h1>  
<p>My first paragraph.</p>  
  
<script>  
window.alert(5 + 6);  
</script>  
</body>  
</html>

Trong trường hợp gỡ rối, có thể sử dụng console.log để hiển thị dữ liệu, thuận tiện cho việc gỡ rối.

<html>  
<body>  
<script>  
console.log(5 + 6);  
</script>  
</body>  
</html>

Đối với trình duyệt Chrome, có thể bấm F12 để để mở chế độ debug và vào tab Console để xem kết quả in ra từ câu lệnh log.

#### JS Statements

Các câu lệnh trong JavaScript gồm gồm có các giá trị, toán tử, biểu thức, từ khoá và các chú thích. Các câu lệnh trong JavaScript thường được viết trên một dòng. Dấu ; đặt ở cuối mỗi câu lệnh để phân tách các câu lệnh. JavaScript bỏ qua các đoạn có nhiều dấu khoảng trắng, những đoạn có nhiều dấu khoảng trắng có thể thêm vào chỉ để cho mã lệnh dễ nhìn hơn. Cũng để thuận tiện hơn cho việc đọc, các lập trình viên thường tránh những câu lệnh quá dài, hoặc để nhiều câu lệnh trên một dòng. Thông thường không nên để quá 80 kỹ tự trên một dòng, trường hợp dòng quá dài có thể ngắt ngay sau các toán tử, ví dụ:

document.getElementById("demo").innerHTML =  
"Hello Dolly!";

JavaScript nhóm các câu lệnh thành các khối mã trong các dấu ngoặc {…}. Mục đích của các khối mã này dùng để định nghĩa các câu lệnh sẽ được thực thi cùng với nhau.

Trong JavaScript các câu lệnh thường được bắt đầu với các Từ khoá (keyword), các từ khoá là các từ dành riêng, và không được sử dụng cho việc đặt tên hàm hay biến. Cách sử dụng của một số Từ khoá thông dụng sẽ được trình bày trong các phần sau. Các Từ khoá hay dùng bao gồm:

|  |  |
| --- | --- |
| Từ khoá | Mô tả |
| break | Kết thúc câu lệnh switch hoặc loop |
| continue | Bỏ qua một lượt lặp trong vòng lặp |
| debugger | Dừng thực thi câu lệnh, và gọi hàm gỡ rối |
| do ... while | Thực thi một khối lệnh và lặp lại các câu lệnh trong khối lệnh này cho tới khi điều kiện còn đúng |
| function | Khai báo một hàm |
| if ... else | Thực thi khối lệnh khi điều kiện đúng |
| return | Thoát khỏi hàm |
| switch | Lựa chọn một khối lệnh để thực thi phụ thuộc vào điều kiện |
| try ... catch | Bắt lỗi của một khối lệnh |
| var | Khai báo biến |

#### JS Comments

Để viết các chú thích trong khi lập trình JavaScript có thể sử dụng // hoặc /\* \*/

Trong đó để ghi chú thích trên một dòng ta sử dụng // như ví dụ sau:

var x = 5;      // Declare x, give it the value of 5  
var y = x + 2;  // Declare y, give it the value of x + 2

Trường hợp cần ghi chú thích trên nhiều dòng, ta sử dụng /\* để bắt đầu và kết thúc bằng \*/ như ví dụ sau:

/\*  
The code below will change  
the heading with id = "myH"  
and the paragraph with id = "myP"  
in my web page:  
\*/  
document.getElementById("myH").innerHTML = "My First Page";  
document.getElementById("myP").innerHTML = "My first paragraph.";

Ngoài tác dụng ghi chú thích, sử dụng // hoặc /\* \*/ với các khối lệnh để ngăn việc thực thi các khối lệnh đó

#### JS Variables

Biến được sử dụng để lưu chữ giá trị, để khai báo biến trong JavaScript ta sử dụng từ khoá var.

var x = 5;  
var y = 6;  
var z = x + y;

Việc đặt tên biến cần được đặt tên theo quy tắc:

* Không đặt tên biến trùng với từ khoá
* Tên biến có thể chứa số, chữ cái hoặc dấu gạch dưới (\_), hoặc ký tự đô la ($)
* Ký tự đầu tiên phải là chữ hoặc dấu gạch dưới (\_), hoặc ký tự đô la ($)
* JavaScript phân biệt chữ hoa và chữ thường, total hay Total được xem là 2 biên khác nhau.

Toán tử gán (=) được sử dụng để gán giá trị cho một biến, một biểu thức x = x + 5, trong JavaScript có nghĩa là cộng thêm 5 vào biến x.

Kiểu dữ liệu trong JavaScript có 2 kiểu phổ biển là kiểu số và kiểu chuỗi. Trong đó kiểu chuỗi lưu trữ chuỗi ký tự, và chuỗi kỹ tự được đặt trong dấu “”. Kiểu số thì không cần sử dụng dấu “”. Ví dụ:

var pi = 3.14;  
var person = "John Doe";  
var answer = 'Yes I am!';

Để tạo một biến trong JavaScript cần phải khai báo, như ví dụ trên, để khai báo biến ta sử dụng từ khoá var. Sau khi biến được khai báo có thể gán giá trị cho biến bằng cách sử dụng toán tử gán. Việc gán gía trị có thể thực hiện ngay khi khai báo biến hoặc sau khi khai báo biến. Trong ví dụ dưới đây, chúng ta tạo một biến tên là carName và gán giá trị Volvo cho nó, sau đó in giá trị đó ra màn hình.

<p id="demo"></p>  
<script>  
var carName = "Volvo";  
document.getElementById("demo").innerHTML = carName;   
</script>

Việc khai báo và gán giá trị cho nhiều biến có thể được thực hiện trên một dòng, ví dụ:

var person = "John Doe", carName = "Volvo", price = 200;

Trong trường hợp khai báo lại một biến, biến đó sẽ không bị mất giá trị, ví dụ:

var carName = "Volvo";  
var carName;

#### JS Operators

Trong JavaScript có nhiều toán tử, bao gồm toán tử gán, toán tử toán học, toán tử với chuỗi, toán tử so sánh, toán tử logic, toán tử kiểm tra kiểu, toán tử bit.

Các toán tử toán học bao gồm:

|  |  |
| --- | --- |
| Toán tử | Mô tả |
| + | Cộng |
| - | Trừ |
| \* | Nhân |
| \*\* | Luỹ thừa |
| / | Chia |
| % | Chia lấy phần dư |
| ++ | Tăng |
| -- | Giảm |

Ví dụ với toán tử +:

var x = 100 + 50;

var x = a + b;

Chú ý đối với phép chia lấy phần dư, ta sử dụng toán tử %. Ví dụ:

var x = 5;  
var y = 2;  
var z = x % y;

Ví dụ trên sẽ cho kết quả bằng 1.

Các toán tử tăng (++) và giảm (--) được sử dụng khi chúng ta cần tăng, hoặc giảm giá trị của biến 1 đơn vị. Ví dụ sau sẽ thêm 1 vào biến x:

var x = 5;  
x++;

Với toán tử luỹ thừa, ta có thể sử dụng để tính luỹ thừa của một số, ví dụ muốn tính lập phương của 5, ta có thể viết:

var x = 5;  
var z = x \*\* 3;

Toán tử luỹ thừa tương đương với việc sử dụng hàm Math.pow(x,y). Ví dụ với việc tính lập phương của 5 ta cũng có thể viết Math.pow(5, 3)

Toán tử gán:

|  |  |  |
| --- | --- | --- |
| Toán tử | Ví dụ | Tương đương với |
| = | x = y | x = y |
| += | x += y | x = x + y |
| -= | x -= y | x = x - y |
| \*= | x \*= y | x = x \* y |
| /= | x /= y | x = x / y |
| %= | x %= y | x = x % y |

Toán tử gán cũng có thể thay đổi giá trị của một biến như trong trường hợp Toán tử gán += được sử dụng để cộng thêm giá trị vào một biến. Ví dụ muốn cộng thêm 5 vào biến x, ta có thể viết:

var x = 10;  
x += 5;

Toán tử so sánh:

|  |  |
| --- | --- |
| Toán tử | Mô tả |
| == | equal to |
| === | equal value and equal type |
| != | not equal |
| !== | not equal value or not equal type |
| > | greater than |
| < | less than |
| >= | greater than or equal to |
| <= | less than or equal to |
| ? | ternary operator |

Toán tử kiểu:

|  |  |
| --- | --- |
| Toán tử | Mô tả |
| typeof | Returns the type of a variable |
| instanceof | Returns true if an object is an instance of an object type |

Toán tử bit

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Toán tử | Mô tả | Ví dụ | Tương đương với | Kết quả | Số thập phân |
| & | AND | 5 & 1 | 0101 & 0001 | 0001 | 1 |
| | | OR | 5 | 1 | 0101 | 0001 | 0101 | 5 |
| ~ | NOT | ~ 5 | ~0101 | 1010 | 10 |
| ^ | XOR | 5 ^ 1 | 0101 ^ 0001 | 0100 | 4 |
| << | Zero fill left shift | 5 << 1 | 0101 << 1 | 1010 | 10 |
| >> | Signed right shift | 5 >> 1 | 0101 >> 1 | 0010 | 2 |
| >>> | Zero fill right shift | 5 >>> 1 | 0101 >>> 1 | 0010 | 2 |

#### JS Data Types

JavaScript có 3 kiểu dữ liệu chính là Số, Chuỗi kỹ tự, và Đối tượng. Ví dụ:

var length = 16;                               // Số  
var lastName = "Johnson";                      // Chuỗi ký tự  
var x = {firstName:"John", lastName:"Doe"};    // Đối tượng

Khi sử dụng var để khai báo biến, kiểu dữ liệu là động. Điều đó đồng nghĩa với việc một biến có thể lưu trữ nhiều kiểu dữ liệu tại các thời điểm khác nhau. Ví dụ biến x với khai báo sau có thể nhận giá trị là số, hoặc giá trị là chuỗi ký tự:

var x;           // Now x is undefined  
x = 5;           // Now x is a Number  
x = "John";      // Now x is a String

Kiểu dữ liệu rất quan trong khi lập trình với JavaScript. Khi ta thực hiện một phép tính:

var x = 16 + "Volvo";

JavaScript sẽ hiểu là “16” + “Volvo” và kết quả sẽ là 16Volvo. Tuy nhiên trong trường hợp ta thực hiện phép tính:

var x = 16 + 4 + "Volvo";

JavaScript sẽ thực hiện từ trái sang phải và kết quả sẽ được tính là (16 + 4 ) + “Volvo” = “20Volvo”. Đối với trường hợp:

var x = "Volvo" + 16 + 4;

Vì thứ tự thực hiện từ trái sang phải, kết quả sẽ được tính tương đương với (“Volvo” + 16) + 4 = “Volvo164”.

Đối với mảng, có thể sử dụng ngặoc vuông [] để khai báo, ví dụ:

var cars = ["Saab", "Volvo", "BMW"];

Với kiểu đối tượng, một đối tượng trong JavaScript được đặt trong dấu ngoặc nhọn {}, Các thuộc tính của đối tượng được viết theo dạng cặp phân cách bởi dấu hai chấm :. Các thuộc tính được phân cách bởi dấu phảy (,). Ví dụ để tạo ra đối tượng person, có các thuộc tính firsName, lastName, age và eyeColor :

var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};

Để kiểm tra kiểu của một biến tại một thời điểm, có thể sử dụng toán tử typeof, ví dụ:

typeof ""                  // Trả về "string"  
typeof "John"              // Trả về "string"  
typeof 3.14           // Trả về "number"

Trường hợp khi khai báo một biến với var mà chưa gán giá trị cho biến đó thì kiểu của biến sẽ là undefined.

Trong JavaScript null có nghĩa là “không có gì”. Null được sử dụng khi ta muốn thiết lập một đối tượng nhận giá trị “không có gì” hay “rỗng”. Ví dụ:

var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};  
person = null;        // Now value is null, but type is still an object

Chúng ta cũng có thể đặt một object = undefined. Sự khác nhau giữa null và undefine thể hiện trong ví dụ sau:

typeof undefined           // undefined  
typeof null                // object  
null === undefined         // false  
null == undefined          // true

#### JS Functions

Một hàm là một khối mã lệnh được thiết kế để thực hiện một công việc cụ thể nào đó. Một hàm trong JavaScript được khai báo bằng từ khoá function, tiếp theo là tên và dấu () chứa các tham số. Các tham số truyền vào hàm, được đặt trong dấu () cà được truyền vào khi gọi hàm. Một hàm có giá trị trả về thông qua lệnh return. Khi gặp lệnh return hàm cũng kết thúc việc thực thi và giá trị sẽ được trả về nơi hàm được gọi. Ví dụ giá trị trả về của hàm trong trường hợp sau là 12:

var x = myFunction(4, 3);    // Gọi hàm, giá trị trả về lưu x  
function myFunction(a, b) {  
    return a \* b;            // Giá trị trả về là tích của a và b  
}

Lưu ý các biến được khai báo trong hàm được gọi là các biến địa phương, và chỉ có thể được truy cập bên trong của hàm. Chính vì các biến chỉ có thể truy cập bên trong hàm ta có thể sử dụng các biến cùng tên trong các hàm khác nhau. Các biến địa phương được tạo và sử dụng khi hàm bắt đầu thực thi, và sẽ được xoá khi hàm kết thúc.

#### JS Objects

Đối tượng được sử dụng nhiều trong JavaScript. Mỗi một đối tượng sẽ bao gồm các thuộc tính và phương thức. Ví dụ đối car là một đối tượng, khi đó:

|  |  |  |
| --- | --- | --- |
| Đối tượng | Thuộc tính | Methods |
| https://www.w3schools.com/js/objectExplained.gifcar | car.name = Fiat car.model = 500 car.weight = 850kg car.color = white | car.start() car.drive() car.brake() car.stop() |

Có thể thấy các xe đều có các thuộc tính tương tự nhau, nhưng giá trị khác nhau. Các xe cũng có các phương thức tương tự nhau nhưng sẽ thực thi khác nhau. Trong JavaScript để tạo ra một đối tượng ta sẽ gán các giá trị khởi tạo cho đối tượng đó. Ví dụ với xe Fiat, nặng 500kg, màu trắng sẽ được tạo đối tượng như sau:

var car = {type:"Fiat", model:"500", color:"white"};

Để dễ đọc, các thuộc tính thường được khai báo trên các dòng khác nhau, ví dụ:

var person = {  
    firstName:"John",  
    lastName:"Doe",  
    age:50,  
    eyeColor:"blue"  
};

Để truy cập vào các thuộc tính của đối tượng, có thể sử dụng dấu chấm (.) hoặc dấu ngoặc vuông []

Ví dụ để truy cập vào thuộc tính lastName của đối tượng person, có thể viết:

person.lastName;

hoặc

person["lastName"];

Các đối tượng cũng có thể có các phương thức (method). Các phương thức được xem là các hành động (action) mà đối tượng có thể thực hiện. Việc khai báo phương thức tương tự như việc khai báo hàm. Ví dụ đối tượng person có phương thức fullname trả về họ tên đầy đủ:

var person = {  
    firstName: "John",  
    lastName : "Doe",  
    id       : 5566,  
    fullName : function() {  
        return this.firstName + " " + this.lastName;  
    }  
};

Khi gọi phương thức trên, sử dụng cú pháp objectName.methodName(). Ví dụ:

name = person.fullName();

Trường hợp viết phương thức mà thiếu dấu (). Chương trình sẽ cho ra định nghĩa (các câu lện) của hàm đó. Ví dụ:

name = person.fullName;

Sẽ trả về:

function() { return this.firstName + " " + this.lastName; }

#### JS Events

Một sự kiện là một hành động nào đó tác động lên các thành phần HTML mà tác có thể bắt các sự kiện này và yêu cầu JavaScript thực hiện các câu lệnh khi một sự kiện diễn ra. Để gán sự kiện cho cho một thành phần HTML, sử dụng cú pháp sau đây:

<element event='some JavaScript'>

Ví dụ khi thêm sự kiện bấm cho một nút bấm:

<button onclick="document.getElementById('demo').innerHTML = Date()">The time is?</button>

Trong ví dụ trên, JavaScript sẽ thay đổi nội dung của phần tử HTML có id là demo. Trường hợp muốn thực hiện nhiều đoạn mã lệnh khi một sự kiện diễn ra, người ta thường gọi hàm. Ví dụ như sau:

<button onclick="displayDate()">The time is?</button>

Các sự kiện phổ biến trong JavaScript:

|  |  |
| --- | --- |
| Sự kiện | Mô tả |
| onchange | Khi một phần tử HTML thay đổi |
| onclick | Khi người dùng bấm lên một phần tử HTML |
| onmouseover | Khi người dùng di chuyển chuột trên một phần tử HTML |
| onmouseout | Khi người dùng di chuyển chuột khỏi một phần tử HTML |
| onkeydown | Khi người dùng bấm bàn phím |
| onload | Khi trình duyệt kết thúc việc tải trang |

Với việc điều khiển sự kiện, JavaScript có thể điều khiển, xác thực dữ liệu người dùng nhập vào hoặc thực hiện các hành động của người dùng cũng như của trình duyệt. Ví dụ như:

* Những thứ cần làm mỗi khi tải trang
* Những thứ cần làm mỗi khi trang bị đóng.
* Hành động cần thực hiện khi người dùng bấm, di chuyển chuột.
* Nội dung cần xác thực khi người dùng nhập dữ liệu
* …

#### JS Strings

JavaScript sử dụng chuỗi để lưu trữ và thao tác với ký tự và văn bản. Một chuỗi có thể không chứa, hoặc có chứa các ký tự, à được đặt trong các dấu nháy kép “” hoặc nháy đơn ‘’. Ví dụ:

var driverName = "John Doe";

var carName = "Toyota Camry";

Chúng ta có thể để các dấu nháy kép, và nháy đơn bên trong chuỗi. Ví dụ:

var answer = "It's alright";  
var answer = "He is called 'Johnny'";  
var answer = 'He is called "Johnny"';

Độ dài của chuỗi được xác định thông qua thuộc tính property. Ví dụ:

var txt = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";  
var sln = txt.length;

Trong chuỗi khi chúng ta sử dụng các ký tự đặc biệt, nên sử dụng thêm dấu \. Đặc biệt trong trường hợp với chuỗi có các ký tự nháy đơn và nháy kép. Ví dụ:

var x = "We are the so-called \"Vikings\" from the north.";

var x = 'It\'s alright.';

Trong trường hợp muốn thêm chính ký tự \ vào trong chuỗi, cần sử dụng \\.

Một chuỗi có thể một đối tượng, chính vì vậy để tạo một chuỗi có thể sử dụng một trong 2 cú pháp sau:

var x = "John";  
var y = new String("John");

Tuy nhiên cần lưu ý khi sử dụng toán tử typeof để kiểm tra kiểu của x thì với 2 cách trên sẽ trả về 2 kết quả khác nhau. Sử dụng var x = “John” thì kiểu của x sẽ là string, trong khi nếu sử dụng var y = new String(“John”) thì kiểu của y sẽ là object. Và trong trường hợp sử dụng phép so sánh (x==y) thì kết quả trả về sẽ là false do 2 biến khác nhau về kiểu dữ liệu. Tuy nhiên cũng cần lưu ý rằng nếu các 2 biến đều khai báo là đối tượng thì việc so sánh (x == y) cũng sẽ trả về false vì x và y là 2 đối tượng khác nhau.

Để so sánh 2 chuỗi ta cần sử dụng phương thức valueOf(). Ví dụ khi so sánh x == y.valueOf() sẽ trả về true:

#### JS String Methods

Đối tượng chuỗi có nhiều phương thức hữu ích. Ví dụ để tìm một chuỗi trong một chuỗi có thể sử dụng phương thức indexOf(). Phương thức này sẽ trả về vị trí ký tự đầu tiên tìm thấy. Ví dụ để tìm ký tự locate xuất hiện ở vị trí nào trong chuỗi:

var str = "Please locate where 'locate' occurs!";  
var pos = str.indexOf("locate"); // trả về 7

Lưu ý JavaScript đánh vị trí từ 0, trong trường hợp trên sẽ trả về kết quả là 7.

Trường hợp muốn tìm ngược từ dưới có thể sử dụng phương thức lastIndexOf(). Ví dụ:

var str = "Please locate where 'locate' occurs!";  
var pos = str.lastIndexOf("locate");

Lưu ý cả 2 hàm indexOf và lastIndexOf đều trả về -1 trong trường hợp không tìm thấy. Cả hai hàm trên đều có thể truyền thêm tham số thứ 2 để xác định vị trí bắt đầu tìm kiếm:

var str = "Please locate where 'locate' occurs!";  
var pos = str.indexOf("locate",15);

Ngoài ra có thể sử dụng hàm search() để tìm kiếm chuỗi. Hàm search cho phép sử dụng tìm kiếm với biểu thức nguyên mẫu.

Để trích xuất chuỗi, có thể sử dụng các hàm sau đây:

* Slice(start, end)
* Substring(start, end)
* Substr(start, length).

Chú ý phương thức slice(start, end) và substring(start, end) có một số điểm giống và khác nhau.

Để thay thế nội dung của các ký tự trong chuỗi có thể sử dụng phương thức replace(). Ví dụ sau đây sẽ thay thế chuỗi Microsoft bằng Google :

str = "Please visit Microsoft!";  
var n = str.replace("Microsoft", "Google");

Lưu ý hàm replace chỉ thay thế chuỗi ký tự đầu tiên nó tìm thấy, để thay đổi toàn bộ phải sử dụng đến biểu thức nguyên mẫu. Biểu thức nguyên mẫu có nhiều lợi thế trong việc tìm kiếm cũng như thay thế. Ví dụ với biểu thức nguyên mẫu với /i để thay thế khi tìm kiếm cả chữ hoa và chữ thường:

str = "Please visit Microsoft!";  
var n = str.replace(/MICROSOFT/i, "Google"

Hay tìm kiếm trong toàn bộ thì sử dụng thêm /g (g là viết tắt của global)

str = "Please visit Microsoft and Microsoft!";  
var n = str.replace(/Microsoft/g, "W3Schools");

Biểu thức nguyên mẫu có nhiều nội dung khác, và sẽ được đề cập trong phần sau.

Ngoài ra với chuỗi ký tự có thể sử dụng nhiều các phương hữu ích khác. Ví dụ như phương thức toUpperCase() để chuyển từ cthường sang chữ hoa, hay phương tức toLowerCase() để chuyển từ chữ hoa sang chữ thường.

Phương thức concat() được sử dụng để nối 2 chuỗi ký tự. Ví dụ:

var text1 = "Hello";  
var text2 = "World";  
var text3 = text1.concat(" ", text2);

Chú ý rằng việc nối chuỗi cùng có thể được thực hiện với toán tử +.

Phương thức trim() được sử dụng để xoá các khoảng trắc ở đầu và cuối chuỗi. Ví dụ sau hàm trim sẽ xoá các khoảng trắng ở đầu và cuối của chuỗi ký tự đã cho:

var str = "       Hello World!        ";  
alert(str.trim());

Để lấy một ký tự cụ thể trong chuỗi có thể sử dụng phương thức charAt(), hoặc lấy mã của ký tự bằng cách sử dụng phương thức charCodeAt(). Ví dụ:

var str = "HELLO WORLD";  
str.charAt(0);            // trả về H

str.charCodeAt(0);         // Trả về 72

Trong một số trường hợp cần tách chuỗi, hoặc chuyển chuỗi về một mảng, có thể sử dụng phương thức split(). Phương thức này sẽ cần truyền vào tham số là ký tự nhận biết để tách chuỗi. Ví dụ:

var txt = "a,b,c,d,e";   // String  
txt.split(",");          //   
txt.split(" ");          //   
txt.split("|");          //

Trường hợp muốn chuyển một chuỗi thành một mảng với các phần tử mảng lần lượt là các ký tự của chuỗi thì không cần chuyền tham số cho phương thức split() này. Ví dụ:

var txt = "Hello";       // String  
txt.split("");           // Split in characters

#### JS Numbers

JavaScript chỉ có một kiểu dữ liệu số. Một số có thể có phần thập phân hoặc không. Ví dụ có thể khai báo biến số như sau:

var x = 3.14;    // A number with decimals  
var y = 3;       // A number without decimals

Số cũng có thể được biểu diễn dạng mũ e như sau:

var x = 123e5;    // 12300000  
var y = 123e-5;   // 0.00123

Không giống như hầu hết ngôn ngữ lập trình khác JavaScript không định nghĩa các kiểu dữ liệu số khác nhau. Tất cả các số trong JavaScript đều là số thực và sử dụng 64 bit. Vơi 64 bit một số thực trong JavaScript sẽ sử dụng 52 bits cho phần giá trị, 11 bit cho phần số mũ và 1 bit cho dấu. Độ chính xác của số trong JavaScript là 15 chữ số. Ví dụ:

var x = 999999999999999;   // x will be 999999999999999  
var y = 9999999999999999;  // y will be 10000000000000000

Phần thập phân có độ chính xác 17 chữ số

Một chuỗi trong JavaScript có thể có chứa nội dung là một số. Như vậy có thể khai báo một số như một chuỗi ký tự. Ví dụ:

var x = "100";  
var y = "10";  
var z = x / y;       // z will be 10

Với cách khai báo như trên, nếu thực hiện phép chia x cho y, JavaScript sẽ cố gắng thực hiện chuyển từ chuỗi thành số khi sử dụng toán tử toán học trên các chuỗi số đó. Tuy nhiên điều này chỉ đúng với các toán tử nhân (\*), chia (/) hoặc trừ (-). Với toán tử cộng, JavaScript sẽ hiểu đó là ghép chuỗi. Ví dụ:

var x = "100";  
var y = "10";  
var z = x + y;       // z sẽ nhận kết quả là “10010” thay vì 110

Trường hợp thực hiện các toán tử toán học, mà x hoặc y không thể chuyển sang dạng số. JavaScript sẽ trả về kết quả là NaN (Not a Number – không phải là số). Ví dụ:

var x = 100 / "Apple";  // x sẽ là NaN

Trong những trường hợp này, có thể kiểm tra xem x có phải là NaN hay không bằng cách sử dụng hàm isNaN(x). Ví dụ:

var x = 100 / "Apple";  
isNaN(x);               // trả về true vì x là NaN

Các số trong JavaScript cũng thể nhân giá trị vô cùng (Infinity) hoặc âm vô cùng -Infinity trong trường hợp số đó vượt quá khoảng lưu trữ số của JavaScript. Ví dụ khi thực hiện phép chia cho 0. Kết quả sẽ trả về là dương vô cùng hoặc âm vô cùng.

var x =  2 / 0;          // x bằng Infinity  
var y = -2 / 0;          // y bằng -Infinity

Trong JavaScript cho biểu diễn số dưới dạng Hexadecimal (hệ thập lục phân) bằng cách sử dụng tiền tố 0x. Ví dụ:

var x = 0xFF;           // x có giá trị 255 trong hệ thập phân.

Một số có thể hiển thị bằng cách sử dụng hệ 2 8 10 hay 16 bằng cách sử dụng hàm toString() với tham số tương ứng. Ví dụ:

var myNumber = 32;  
myNumber.toString(10);  // trả về 32  
myNumber.toString(32);  // trả về 10  
myNumber.toString(16);  // trả về 20  
myNumber.toString(8);   // trả về 40  
myNumber.toString(2);   // trả về 100000

Số trong JavaScript cũng giống như chuỗi có thể khai báo như là một đối tượng với từ khoá new như sau:

var x = new Number(123);

Khi đó x sẽ có kiểu là object. Tuy nhiên đây là cách sử dụng không phổ biến, không cần thiết và có thể làm chậm hệ thống. với kiểu khai bảo thông thường, vẫn có thể sử dụng được toàn bộ các phương thức của Numbers. Một số phương thức hữu dụng hay dùng như là toString(), to Expoential(), toFixed(), toPrecision(), valueOf()… hay các hằng như MIN\_VALUE, MAX\_VALUE, POSITIVE\_INFINITY, NEGATIVE\_INFINITY.

Để chuyển đổi một kiểu dữ liệu sang số có thể sử dụng các phương thức Number(), parseInt(), hoặc parseFloat(). Ví với phương thức Number():

Number(true);          // returns 1  
Number(false);         // returns 0  
Number("10");          // returns 10  
Number("  10");        // returns 10  
Number("10  ");        // returns 10  
Number(" 10  ");       // returns 10  
Number("10.33");       // returns 10.33  
Number("10,33");       // returns NaN  
Number("10 33");       // returns NaN   
Number("John");        // returns NaN

Phương thức number chuyển từ ngày tháng sang số, ví dụ:

Number(new Date("2017-09-30"));    // Trả về 1506729600000

Hàm parseInt() sử dụng để chuyển từ chuỗi về số. Phương thứ này không cho phép ký tự dấu cách. Trường hợp có dấu cách, chỉ trả về số đầu tiên. Trường hợp không thể chuyển sang số, hàm này sẽ trả về NaN. Các ví dụ đối với hàm parseInt():

parseInt("10");         // returns 10  
parseInt("10.33");      // returns 10  
parseInt("10 20 30");   // returns 10  
parseInt("10 years");   // returns 10  
parseInt("years 10");   // returns NaN

Trong trường hợp muốn chuyển sang kiểu số thực, có thể sử dụng hàm parseFloat(). Ví dụ với hàm parseFloat():

parseFloat("10");        // returns 10  
parseFloat("10.33");     // returns 10.33  
parseFloat("10 20 30");  // returns 10  
parseFloat("10 years");  // returns 10  
parseFloat("years 10");  // returns NaN

#### JS Arrays

Mảng được sử dụng để lưu trữ nhiều giá trị (thường là cùng kiểu) vào một biến duy nhất. Khai bảo mảng sử dụng dấu ngoặc vuông [], từ khoá new

var cars = ["Saab", "Volvo", "BMW"];

var cars = new Array("Saab", "Volvo", "BMW");

Để truy cập đến từng phần tử của mảng, sử dụng dấu ngoặc vuông [] với phẩn tử của mảng. Chú ý phần tử đầu tiên của mảng bắt đầu từ 0. Ví dụ để truy cập đến phần tử đầu tiên của mảng cars:

var cars = ["Saab", "Volvo", "BMW"];  
cars[0] = "Opel";  
document.getElementById("demo").innerHTML = cars[0];

Mỗi phần tử của mảng cũng có thể là một đối tượng, thậm chí có thể là một mảng khác. Ví dụ:

myArray[0] = Date.now;  
myArray[1] = myFunction;  
myArray[2] = myCars;

Khi thao tác với mảng, người ta thường sử dụng vòng lặp, để đi qua các phần tử của mảng. Ví dụ dùng vòng lặp for để in ra danh sách các phần tử của mảng:

var fruits, text, fLen, i;  
fruits = ["Banana", "Orange", "Apple", "Mango"];  
fLen = fruits.length;  
text = "<ul>";  
for (i = 0; i < fLen; i++) {  
  text += "<li>" + fruits[i] + "</li>";  
}  
text += "</ul>";

Ngoài ra cũng có thể sử dụng phương thức forEach() của mảng. Ví dụ:

var fruits, text;  
fruits = ["Banana", "Orange", "Apple", "Mango"];  
text = "<ul>";  
fruits.forEach(myFunction);  
text += "</ul>";  
function myFunction(value) {  
  text += "<li>" + value + "</li>";  
}

Để thêm các phần tử của mảng, sử dụng phương thức push(). Ví dụ:

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits.push("Lemon");    // adds a new element (Lemon) to fruits

Ngoài ra cũng có thể sử dụng chỉ số, và sử dụng thuộc tính length để truy cập vào vị trí cuối cùng của mảng.

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits[fruits.length] = "Lemon";    // adds a new element (Lemon) to fruits

Mảng là một dạng đặc biệt của đối tượng, và từ khoá typeof cũng se trả về kiểu là “object” với các mảng. Tuy nhiên khi sử dụng theo dạng mảng thường truy cập theo chỉ số. Ví dụ person[0].

var person = ["John", "Doe", 46];

Trong khi với đối tượng thì sẽ khai báo thêm tên thuộc tính và truy cập qua tên. Ví dụ person.firstName sẽ trả về John trong trường hợp sau:

var person = {firstName:"John", lastName:"Doe", age:46};

Mảng có thể khai báo bằng cách sử dụng [] hoặc new. Tuy nhiên, cách sử dụng [] tốt hơn vì tránh nhầm lẫn. Ví dụ để tạo một mảng có 6 phần tử có thể sử dụng 2 cách:

var points = new Array(40, 100, 1, 5, 25, 10); // Bad  
var points = [40, 100, 1, 5, 25, 10];          // Good

Tuy nhiên khi viết như sau thì lại được hiểu tạo ra một mảng có 40 phần tử:

var points = new Array(40);

Do typeof của mảng luôn trả về object, nên để kiểm tra xem có phải là một mảng hay không, có thể sử dụng hàm isArray(). Ví dụ:

Array.isArray(fruits);   // trả về true

Hoặc sử dụng từ khoá instanceof. Ví dụ:

fruits instanceof Array;   // trả về true

#### JS Array Methods

Mảng có nhiều phương thức hữu ích như toString() để chuyển mảng về dạng chuỗi. Ví dụ:

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
document.getElementById("demo").innerHTML = fruits.toString();

Câu lệnh toString sẽ rả về:

Banana,Orange,Apple,Mango

Phương thức join() cũng chuyển một mảng thành chuỗi nhưng có thể khai báo thêm từ phân cách các phần tử. Ví dụ:

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
document.getElementById("demo").innerHTML = fruits.join(" \* ");

Sẽ trả về:

Banana \* Orange \* Apple \* Mango

Phương thức pop() sẽ xoá phần tử cuối cùng của mảng, và trả về phần tử đó của mảng.

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
var x = fruits.pop();

Phuơng thức pop xoá phần tử cuối cùng của mảng trong khi phương thức shift() được sử dụng để xoá phần tử đầu tiên của mảng. Cần lưu ý rằng phương thức shift() sẽ thực hiện dịch các phần tử của mảng, đồng nghĩa với việc các phần tử trong mảng bị thay đổi chỉ sổ. Ví dụ phần tử đang có chỉ số [2] thì sau khi dịch sẽ có chỉ số [1]. Ví dụ:

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits = fruits.shift();            // Removes the first element "Banana" from fruits

Phương thức push() sẽ thêm phần tử mới vào cuối của chuỗi, và trả về mảng mới sau khi đã thêm phần tử.

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits= fruits.push("Kiwi");       //  Adds a new element ("Kiwi") to fruits

Tương tự như phương thức push, phương thức unshift() cũng thêm một phần tử thêm phần tử vào đầu của mảng. Cũng giống phương thức shift(), phương thức này cũng làm thay đổi chỉ số của các phần tử còn lại trong mảng.

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits.unshift("Lemon");    // Adds a new element "Lemon" to fruits

Để thay đổi nội dung của các phần tử trong mảng, cần truy cập vào phần tử của mảng đó thông qua chỉ số. Ví dụ:

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits[0] = "Kiwi";        // Changes the first element of fruits to "Kiwi

Sử dụng từ khoá delete để xoá phần tử của mảng. Tuy nhiên sử dụng delete đôi khi sẽ tạo ra các hố (hole) trong mảng là các phần không gian trong mảng không lưu trữ dữ liệu gì. Trường hợp muốn xoá nên sử dụng shift() hoặc pop()

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
delete fruits[0];           // Changes the first element in fruits to **undefined**

Ngoài ra mảng còn được hỗ trợ rất nhiều các hàm hữu ích khác như:

* splice(): thêm hoặc thay thế phần tử vào vị trí bất kỳ trong mảng.
* concat(): ghép 1 mảng vào một mảng khác.
* slice(): cắt một phần của mảng sang một mảng mới

#### JS Array Sort

JavaScript cung cấp hàm sort để sắp xếp các phần tử trong mảng. Ví dụ để sắp xếp các phần tử của mảng fruits theo thứ tự bảng chữ cái:

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits.sort();        // Sorts the elements of fruits

Hàm reverse() sẽ sắp xếp các phần tử ngược lại so với thứ tự ban đầu. Hàm reverse() có thể được sử dụng cùng với hàm sort() để sắp xếp mảng. Ví dụ:

var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits.sort();        // First sort the elements of fruits   
fruits.reverse();     // Then reverse the order of the elements

Cần lưu ý, hảm sort() sẽ sắp xếp các phần tử của mảng theo thứ tự bảng chữ cái. Đối với các kiểu dữ liệu khác, như muốn sắp xếp ta cần cung cấp cho sort() một hàm so sánh. Ví dụ để sắp xếp cho số, cần viết như sau:

var points = [40, 100, 1, 5, 25, 10];  
points.sort(function(a, b){return a - b});

Hàm sort cũng có thể sử dụng để sắp xêp các phần tử của mảng theo thứ tự ngẫu nhiên bằng cách biến đổi hàm so sánh cung cấp cho sort(). Ví dụ như sau:

var points = [40, 100, 1, 5, 25, 10];  
points.sort(function(a, b){return 0.5 - Math.random()});

Để tìm giá trị lớn nhất và nhỏ nhất các phần tử trong mảng, có thể lợi dụng hàm sắp xếp để sắp xếp mảng theo thứ tự sau đó lấy phần tử ở 2 đầu của mảng sẽ được giá trị lớn nhất hoặc nhỏ nhất tương ứng.

Đối với mảng các đối tượng để sắp xếp cũng cần cung cấp hàm so sánh, tuy nhiên thông thường sẽ sắp xếp theo một thuộc tính nào đó của đối tượng. Ví dụ mảng có chứa các đối tượng là thông tin của các xe ô tô như sau:

var cars = [  
  {type:"Volvo", year:2016},  
  {type:"Saab", year:2001},  
  {type:"BMW", year:2010}  
];

Để sắp xếp theo năm, hàm so sánh được viết như sau:

cars.sort(function(a, b){return a.year - b.year});

Trường hợp sắp xếp theo tên, hàm so sánh có thể được viết như sau:

cars.sort(function(a, b){  
  var x = a.type.toLowerCase();  
  var y = b.type.toLowerCase();  
  if (x < y) {return -1;}  
  if (x > y) {return 1;}  
  return 0;  
});

#### JS Array Iteration

Khi làm việc với mảng, nhiều trường hợp cần duyệt qua tất cả các phần tử của mảng. Có nhiều cách để làm điều đó. Ví dụ có thể sử dụng phương thức forEach(). Ví dụ như sau:

var txt = "";  
var numbers = [45, 4, 9, 16, 25];  
numbers.forEach(myFunction);  
function myFunction(value, index, array) {  
  txt = txt + value + "<br>";   
}

Đối với hàm forEach, ta cần truyền vào một hàm, trong đó truyền vào 3 tham số: value, index, và array tương ứng với giá trị, chỉ số, và chính mảng đó. Khi đó ở trong hàm có thể sử dụng các tham số này để thao thác. Như trong ví dụ trên, hàm myFunction sẽ được gọi nhiều lần và giá trị (value) cũng như chỉ số (index) sẽ được thay đổi trong mỗi lần gọi. Số lần gọi hàm sẽ phụ thuộc vào số phần tử của mảng.

Hàm map() cũng được sử dụng tương tự như hàm forEach() nhưng sẽ tạo ra một mảng mới, trong hàm truyền vào có thể thay đổi giá trị, tuy nhiên không làm ảnh hưởng đến mảng ban đầu. Ví dụ sau tạo ra mảng mới với các phần tử được nhân đôi giá trị:

var numbers1 = [45, 4, 9, 16, 25];  
var numbers2 = numbers1.map(myFunction);  
function myFunction(value, index, array) {  
  return value \* 2;  
}

Phương thức filter() cũng tạo ra mảng mới, nhưng sẽ lựa chọn các phần tử phụ thuộc vào giá trị trả về của hàm truyền vào là đúng hay sai. Ví dụ để tạo ra mảng mới từ mảng ban đầu, và chỉ lấy các phần tử có giá trị lớn hơn 18:

var numbers = [45, 4, 9, 16, 25];  
var over18 = numbers.filter(myFunction);  
function myFunction(value, index, array) {  
  return value > 18;  
}

Phương thức reduce() được sử dụng để đưa một mảng về một giá trị duy nhất. Sử dụng hàm này có thể tính tổng các phần tử của mảng như sau:

var numbers1 = [45, 4, 9, 16, 25];  
var sum = numbers1.reduce(myFunction);  
function myFunction(total, value, index, array) {  
  return total + value;  
}

Hàm every() để kiểm tra toàn bộ các phần tử của mảng có thoả mãn một điều kiện gì đó hay không. Ví dụ để kiểm tra các phần tử của mảng có đều lớn hơn 18 hay không:

var numbers = [45, 4, 9, 16, 25];  
var allOver18 = numbers.every(myFunction);  
function myFunction(value) {  
  return value > 18;  
}

Sử dụng phương thức indexOf() để tìm kiếm các phần tử của mảng, hàm này trả về chỉ số phần tử đầu tiên tìm thấy. Ví dụ để tìm trong mảng có hay không có phần tử có giá trị là “Apple”:

var fruits = ["Apple", "Orange", "Apple", "Mango"];  
var a = fruits.indexOf("Apple");

Chú ý cũng có thể sử dụng phương thức lastIndexOf() khi muốn tìm sẽ tìm theo tứ tự ngược lại.

Đối với trường hợp tìm phần tử đầu tiên thoả mãn một điều kiện gì đó, có thể sử dụng hàm find(). Ví dụ để tìm phần tử đầu tiên trong mảng có giá trị lớn hơn 18:

var numbers = [4, 9, 16, 25, 29];  
var first = numbers.find(myFunction);  
function myFunction(value, index, array) {  
  return value > 18;  
}

Nếu chỉ muốn lấy chỉ số của phần tử đó, có thể thay thế bằng việc sử dụng hàm findIndex() như sau:

var numbers = [4, 9, 16, 25, 29];  
var first = numbers.findIndex(myFunction);  
function myFunction(value, index, array) {  
  return value > 18;  
}

#### JS Dates

Khi làm việc với ngày tháng, JavaScript sẽ sử dụng múi giờ của trình duyệt đang sử dụng và hiển thị ngày tháng dưới dạng đầy đủ như sau:

Fri Dec 21 2018 12:04:33 GMT+0700 (Indochina Time)

Để hiển thị.

#### JS Date Formats

* Ngày nhập JavaScript

Nhìn chung có 3 loại định dạng nhập ngày JavaScript:

Thể loại Thí dụ

Ngày ISO "2015 / 03-25" (Tiêu chuẩn quốc tế)

Ngày ngắn "25/03/2015"

Ngày dài "25 tháng 3 năm 2015" hoặc "25 tháng 3 năm 2015"

Định dạng ISO tuân theo một tiêu chuẩn nghiêm ngặt trong JavaScript.

Các định dạng khác không được xác định rõ và có thể là trình duyệt cụ thể.

* Đầu ra ngày JavaScript

Không phụ thuộc vào định dạng đầu vào, JavaScript sẽ (theo mặc định) ngày đầu ra ở định dạng chuỗi văn bản đầy đủ:

Wed Mar 25 2015 07:00:00 GMT+0700 (Giờ Đông Dương)

* Ngày ISO JavaScript

ISO 8601 là tiêu chuẩn quốc tế để thể hiện ngày và thời gian.

Cú pháp ISO 8601 (YYYY-MM-DD) cũng là định dạng ngày JavaScript ưa thích:

* Ngày ISO (Năm và Tháng)

Ngày ISO có thể được viết mà không chỉ định ngày (YYYY-MM):

var d = new Date("2015-03");

* Ngày ISO (Chỉ năm)

Ngày ISO có thể được viết mà không cần tháng và ngày (YYYY):

var d = new Date("2015");

* Ngày ISO (Ngày-Giờ)

Ngày ISO có thể được viết với thêm giờ, phút và giây (YYYY-MM-DDTHH: MM: SSZ):

var d = new Date("2015-03-25T12:00:00Z");

Ngày và giờ được phân tách bằng chữ T.

Thời gian UTC được xác định bằng chữ in hoa Z.

Nếu bạn muốn sửa đổi thời gian liên quan đến UTC, hãy xóa Z và thêm + HH: MM hoặc -HH: MM thay thế:

* Múi giờ

Khi đặt ngày, mà không chỉ định múi giờ, JavaScript sẽ sử dụng múi giờ của trình duyệt.

Khi nhận được một ngày, mà không chỉ định múi giờ, kết quả được chuyển đổi thành múi giờ của trình duyệt.

Nói cách khác: Nếu ngày / giờ được tạo theo GMT (Giờ trung bình Greenwich), ngày / giờ sẽ được chuyển đổi thành CDT (Giờ ban ngày miền Trung Hoa Kỳ) nếu người dùng duyệt từ miền trung Hoa Kỳ.

* Ngày ngắn JavaScript.

Ngày ngắn được viết với cú pháp "MM / DD / YYYY" như thế này

var d = new Date("03/25/2015");

* CẢNH BÁO!

Trong một số trình duyệt, tháng hoặc ngày không có số 0 đứng đầu có thể gây ra lỗi:

var d = new Date("2015-3-25");

Hành vi của "YYYY / MM / DD" không được xác định.

Một số trình duyệt sẽ cố gắng đoán định dạng. Một số sẽ trả lại NaN.

var d = new Date("2015/03/25");

Hành vi của "DD-MM-YYYY" cũng không được xác định.

Một số trình duyệt sẽ cố gắng đoán định dạng. Một số sẽ trả lại NaN.

var d = new Date("25-03-2015");

* Ngày dài JavaScript.

Ngày dài thường được viết bằng cú pháp "MMM DD YYYY" như thế này:

Thí dụ:

var d = new Date("Mar 25 2015");

* Ngày nhập - Ngày phân tích cú pháp

Nếu bạn có chuỗi ngày hợp lệ, bạn có thể sử dụng Date.parse()phương thức để chuyển đổi nó thành mili giây.

Date.parse() trả về số mili giây giữa ngày và ngày 1 tháng 1 năm 1970:

Thí dụ

var msec = Date.parse("March 21, 2012");  
document.getElementById("demo").innerHTML = msec;

Sau đó, bạn có thể sử dụng số mili giây để chuyển đổi nó thành đối tượng ngày :

Thí dụ

var msec = Date.parse("March 21, 2012");  
var d = new Date(msec);  
document.getElementById("demo").innerHTML = d;

#### JS Date Get Methods

Các phương thức này có thể được sử dụng để lấy thông tin từ một đối tượng ngày:

|  |  |
| --- | --- |
| Method | Description |
| getFullYear() | Get the year as a four digit number (yyyy) |
| getMonth() | Get the month as a number (0-11) |
| getDate() | Get the day as a number (1-31) |
| getHours() | Get the hour (0-23) |
| getMinutes() | Get the minute (0-59) |
| getSeconds() | Get the second (0-59) |
| getMilliseconds() | Get the millisecond (0-999) |
| getTime() | Get the time (milliseconds since January 1, 1970) |
| getDay() | Get the weekday as a number (0-6) |
| Date.now() | Get the time. ECMAScript 5. |

* Phương thức getTime ()

Các getTime()phương thức trả về số mili giây kể từ ngày 01 Tháng 1 năm 1970:

Thí dụ

var d = new Date();  
document.getElementById("demo").innerHTML = d.getTime();

* Phương thức getFullYear ()

Các getFullYear()phương thức trả về năm của một ngày như một số có bốn chữ số:

Thí dụ

var d = new Date();  
document.getElementById("demo").innerHTML = d.getFullYear();

* Phương thức getMonth ()

Các getMonth()phương thức trả về tháng của một ngày như một số (0-11):

Thí dụ

var d = new Date();  
document.getElementById("demo").innerHTML = d.getMonth();

Bạn có thể sử dụng một mảng các tên và getMonth()để trả về tháng như một tên

Trong JavaScript, tháng đầu tiên (tháng 1) là tháng số 0, vì vậy tháng 12 trả về tháng số 11

* Phương thức getDate ()

var d = new Date();  
document.getElementById("demo").innerHTML = d.getDate();

* Các getDate()phương thức trả về ngày của một ngày như một số (1-31):
* Phương thức getHours ()

Các getHours()phương pháp trả về giờ của một ngày như một số (0-23):

Thí dụ

var d = new Date();  
document.getElementById("demo").innerHTML = d.getHours();

* Phương thức getSeconds ()

Các getSeconds()phương pháp trả về giây của một ngày như một số (0-59):

Thí dụ

var d = new Date();  
document.getElementById("demo").innerHTML = d.getSeconds();

* Phương thức getMilliseconds ()

Các getMilliseconds()phương pháp trả về mili giây của một ngày như một số (0-999):

Thí dụ

var d = new Date();  
document.getElementById("demo").innerHTML = d.getMilliseconds();9. Phương thức getDay ()

Các getDay()phương thức trả về các ngày trong tuần của một ngày như một số (0-6):

Thí dụ

var d = new Date();  
document.getElementById("demo").innerHTML = d.getDay();

Trong JavaScript, ngày đầu tuần (0) có nghĩa là "Chủ nhật", ngay cả khi một số quốc gia trên thế giới coi ngày đầu tiên của tuần là "Thứ hai"

* Phương pháp ngày UTC

Phương pháp ngày UTC được sử dụng để làm việc với ngày UTC (ngày theo múi giờ chung):

|  |  |
| --- | --- |
| Method | Description |
| getUTCDate() | Same as getDate(), but returns the UTC date |
| getUTCDay() | Same as getDay(), but returns the UTC day |
| getUTCFullYear() | Same as getFullYear(), but returns the UTC year |
| getUTCHours() | Same as getHours(), but returns the UTC hour |
| getUTCMilliseconds() | Same as getMilliseconds(), but returns the UTC milliseconds |
| getUTCMinutes() | Same as getMinutes(), but returns the UTC minutes |
| getUTCMonth() | Same as getMonth(), but returns the UTC month |
| getUTCSeconds() | Same as getSeconds(), but returns the UTC seconds |

* Hoàn thành tham chiếu ngày JavaScript

Tham chiếu chứa các mô tả và ví dụ về tất cả các thuộc tính và phương thức Date

#### JS Date Set Methods

Đặt phương thức Ngày cho phép bạn đặt giá trị ngày (năm, tháng, ngày, giờ, phút, giây, mili giây) cho Đối tượng ngày.

* Đặt phương thức ngày

Đặt phương thức Ngày được sử dụng để đặt một phần của ngày:

|  |  |
| --- | --- |
| Method | Description |
| setDate() | Set the day as a number (1-31) |
| setFullYear() | Set the year (optionally month and day) |
| setHours() | Set the hour (0-23) |
| setMilliseconds() | Set the milliseconds (0-999) |
| setMinutes() | Set the minutes (0-59) |
| setMonth() | Set the month (0-11) |
| setSeconds() | Set the seconds (0-59) |
| setTime() | Set the time (milliseconds since January 1, 1970) |

* Phương thức setFullYear ()

Các setFullYear()phương pháp thiết lập năm của một đối tượng ngày. Trong ví dụ này đến năm 2020:

Thí dụ

<script>  
var d = new Date();  
d.setFullYear(2020);  
document.getElementById("demo").innerHTML = d;  
</script>

* Phương thức setMonth ()

Các setMonth()phương pháp thiết lập tháng của một đối tượng ngày (0-11):

Thí dụ:

<script>  
var d = new Date();  
d.setMonth(11);  
document.getElementById("demo").innerHTML = d;  
</script>

* Phương thức setDate ()

Các setDate()phương pháp thiết lập ngày của một đối tượng ngày (1-31):

Thí dụ:

<script>  
var d = new Date();  
d.setDate(15);  
document.getElementById("demo").innerHTML = d;  
</script>

Các setDate()phương pháp cũng có thể được sử dụng để thêm ngày đến một ngày:

Thí dụ

<script>  
var d = new Date();  
d.setDate(d.getDate() + 50);  
document.getElementById("demo").innerHTML = d;  
</script>

* Phương thức setHours ()

Các setHours()phương pháp đặt giờ của một đối tượng ngày (0-23):

Thí dụ:

<script>  
var d = new Date();  
d.setHours(22);  
document.getElementById("demo").innerHTML = d;  
</script>

* Phương thức setMinutes ()

Các setMinutes()phương pháp lập biên bản một đối tượng ngày (0-59):

Thí dụ:

<script>  
var d = new Date();  
d.setMinutes(30);  
document.getElementById("demo").innerHTML = d;  
</script>

* Phương thức setSeconds ()

Các setSeconds() phương pháp đặt giây của một đối tượng ngày (0-59):

Thí dụ

<script>  
var d = new Date();  
d.setSeconds(30);  
document.getElementById("demo").innerHTML = d;  
</script>

* So sánh ngày

Ngày có thể dễ dàng được so sánh.Ví dụ sau đây so sánh ngày hôm nay với ngày 14 tháng 1 năm 2100:

Thí dụ

var today, someday, text;  
today = new Date();  
someday = new Date();  
someday.setFullYear(2100, 0, 14);  
if (someday > today) {  
  text = "Today is before January 14, 2100.";  
} else {  
  text = "Today is after January 14, 2100.";  
}  
document.getElementById("demo").innerHTML = text;

#### JS Math

* Math.round ()

Math.round(x) trả về giá trị của x được làm tròn đến số nguyên gần nhất của nó:

Thí dụ:

Math.round(4.7);    // returns 5  
Math.round(4.4);    // returns 4

* Math.pow ()

Math.pow(x, y) trả về giá trị của x cho lũy thừa của y:

Thí dụ

Math.pow(8, 2);      // returns 64

* Toán.sqrt ()

Math.sqrt(x) trả về căn bậc hai của x:

Thí dụ

Math.sqrt(64);      // returns 8

* Toán.abs ()

Math.abs(x) trả về giá trị tuyệt đối (dương) của x:

Thí dụ:

Math.abs(-4.7);     // returns 4.7

* Toán.ceil ()

Math.ceil(x) trả về giá trị của x làm tròn lên đến số nguyên gần nhất của nó:

Thí dụ:

Math.ceil(4.4);     // returns 5

* Math.floor ()

Math.floor(x) trả về giá trị của x được làm tròn xuống số nguyên gần nhất của nó:

Thí dụ

Math.floor(4.7);    // returns 4

* Toán học ()

Math.sin(x)trả về sin (giá trị từ -1 đến 1) của góc x (được tính bằng radian).

Nếu bạn muốn sử dụng độ thay vì radian, bạn phải chuyển đổi độ sang radian:

Góc tính theo radian = Góc tính theo độ x PI / 180.

Thí dụ

Math.sin(90 \* Math.PI / 180);     // returns 1 (the sine of 90 degrees)

* Math.cos ()

Math.cos(x) trả về cosin (giá trị từ -1 đến 1) của góc x (được tính bằng radian).

Nếu bạn muốn sử dụng độ thay vì radian, bạn phải chuyển đổi độ sang radian:

Góc tính theo radian = Góc tính theo độ x PI / 180.

Thí dụ:

Math.cos(0 \* Math.PI / 180);     // returns 1 (the cos of 0 degrees)

* Math.min () và Math.max ()

Math.min() và Math.max() có thể được sử dụng để tìm giá trị thấp nhất hoặc cao nhất trong danh sách các đối số:

Thí dụ:

Math.min(0, 150, 30, 20, -8, -200);  // returns -200

* Toán.random ()

Math.random() trả về một số ngẫu nhiên trong khoảng từ 0 (đã bao gồm) và 1 (độc quyền):

Thí dụ:

Math.random();     // returns a random number

* Thuộc tính toán học (hằng số)

JavaScript cung cấp 8 hằng số toán học có thể được truy cập bằng đối tượng Math:

Thí dụ

Math.E // returns Euler's number

Math.PI // returns PI

Math.SQRT2 // returns the square root of 2

Math.SQRT1\_2 // returns the square root of 1/2

Math.LN2 // returns the natural logarithm of 2

Math.LN10 // returns the natural logarithm of 10

Math.LOG2E // returns base 2 logarithm of E

Math.LOG10E // returns base 10 logarithm of E

* Toán xây dựng

Không giống như các đối tượng toàn cầu khác, đối tượng Math không có hàm tạo. Phương thức và tính chất là tĩnh.Tất cả các phương thức và thuộc tính (hằng số) có thể được sử dụng mà không cần tạo đối tượng Math trước.

* Phương pháp đối tượng toán học

|  |  |
| --- | --- |
| Method | Description |
| abs(x) | Returns the absolute value of x |
| acos(x) | Returns the arccosine of x, in radians |
| asin(x) | Returns the arcsine of x, in radians |
| atan(x) | Returns the arctangent of x as a numeric value between -PI/2 and PI/2 radians |
| atan2(y, x) | Returns the arctangent of the quotient of its arguments |
| ceil(x) | Returns the value of x rounded up to its nearest integer |
| cos(x) | Returns the cosine of x (x is in radians) |
| exp(x) | Returns the value of Ex |
| floor(x) | Returns the value of x rounded down to its nearest integer |
| log(x) | Returns the natural logarithm (base E) of x |
| max(x, y, z, ..., n) | Returns the number with the highest value |
| min(x, y, z, ..., n) | Returns the number with the lowest value |
| pow(x, y) | Returns the value of x to the power of y |
| random() | Returns a random number between 0 and 1 |
| round(x) | Returns the value of x rounded to its nearest integer |
| sin(x) | Returns the sine of x (x is in radians) |
| sqrt(x) | Returns the square root of x |
| tan(x) | Returns the tangent of an angle |

#### 2.2.1.23 JS Random

* Toán.random ()

Math.random()trả về một số ngẫu nhiên trong khoảng từ 0 (đã bao gồm) và 1 (độc quyền):

Thí dụ:

Math.random();              // returns a random number

* Số nguyên ngẫu nhiên JavaScript

Math.random() được sử dụng với Math.floor() có thể được sử dụng để trả về số nguyên ngẫu nhiên.

Thí dụ

Math.floor(Math.random() \* 10);     // returns a random integer from 0 to 9

* Một chức năng ngẫu nhiên thích hợp

Như bạn có thể thấy từ các ví dụ trên, có thể là một ý tưởng tốt để tạo một hàm ngẫu nhiên thích hợp để sử dụng cho tất cả các mục đích số nguyên ngẫu nhiên.

Hàm JavaScript này luôn trả về một số ngẫu nhiên giữa min (bao gồm) và max (không bao gồm):

Thí dụ

function getRndInteger(min, max) {  
  return Math.floor(Math.random() \* (max - min) ) + min;  
}

Hàm JavaScript này luôn trả về một số ngẫu nhiên giữa tối thiểu và tối đa (bao gồm cả hai):

#### JS Booleans

Boolean JavaScript đại diện cho một trong hai giá trị: đúng hoặc sai .

* Giá trị Boolean

Rất thường xuyên, trong lập trình, bạn sẽ cần một kiểu dữ liệu chỉ có thể có một trong hai giá trị, như

-CÓ KHÔNG

-BẬT / TẮT

-ĐÚNG SAI

Đối với điều này, JavaScript có kiểu dữ liệu Boolean . Nó chỉ có thể lấy các giá trị đúng hoặc sai .

* Hàm Boolean ()

Bạn có thể sử dụng Boolean()hàm để tìm hiểu xem một biểu thức (hoặc một biến) có đúng không:

Thí dụ:

Boolean(10 > 9)        // returns true

Hoặc thậm chí dễ dàng hơn:

Thí dụ:

(10 > 9)              // also returns true  
10 > 9                // also returns true

* So sánh và điều kiện

Chương so sánh JS cung cấp một cái nhìn tổng quan đầy đủ về các toán tử so sánh.

Chương Điều kiện JS cung cấp một cái nhìn tổng quan đầy đủ về các câu điều kiện.

Dưới đây là một số ví dụ:

|  |  |  |
| --- | --- | --- |
| Operator | Description | Example |
| == | equal to | if (day == "Monday") |
| > | greater than | if (salary > 9000) |
| < | less than | if (age < 18) |

* Mọi thứ với "Giá trị" là Đúng

Ví dụ:

100  
3.14  
-15  
"Hello"  
"false"  
7 + 1 + 3.14

* Mọi thứ không có "Giá trị" là Sai

Giá trị Boolean bằng 0 (không) là sai :

var x = 0;  
Boolean(x);       // returns false

Giá trị Boolean của -0 (trừ 0) là sai :

var x = -0;  
Boolean(x);       // returns false

Giá trị Boolean của "" (chuỗi trống) là sai :

var x = "";  
Boolean(x);       // returns false

Giá trị Boolean của không xác định là sai :

var x;  
Boolean(x);       // returns false

Giá trị Boolean của null là sai :

var x = null;  
Boolean(x);       // returns false

* Booleans có thể là đối tượng

Thông thường booleans JavaScript là các giá trị nguyên thủy được tạo từ chữ:

var x = false;

Nhưng booleans cũng có thể được định nghĩa là các đối tượng với từ khóa new:

var y = new Boolean(false);

Thí dụ

var x = false;  
var y = new Boolean(false);  
// typeof x returns boolean  
// typeof y returns object

#### JS Comparisons

* Toán tử so sánh

Toán tử so sánh được sử dụng trong các câu lệnh logic để xác định sự bằng nhau hoặc khác biệt giữa các biến hoặc giá trị.

Do đó x = 5, bảng dưới đây giải thích các toán tử so sánh:

|  |  |  |  |
| --- | --- | --- | --- |
| Operator | Description | Comparing | Returns |
| == | equal to | x == 8 | false |
| x == 5 | true |
| x == "5" | true |
| === | equal value and equal type | x === 5 | true |
| x === "5" | false |
| != | not equal | x != 8 | true |
| !== | not equal value or not equal type | x !== 5 | false |
| x !== "5" | true |
| x !== 8 | true |
| > | greater than | x > 8 | false |
| < | less than | x < 8 | true |
| >= | greater than or equal to | x >= 8 | false |
| <= | less than or equal to | x <= 8 | true |

* Làm thế nào có thể sử dụng nó

Toán tử so sánh có thể được sử dụng trong các câu lệnh có điều kiện để so sánh các giá trị và thực hiện hành động tùy thuộc vào kết quả:

if (age < 18) text = "Too young";

Bạn sẽ tìm hiểu thêm về việc sử dụng các câu điều kiện trong chương tiếp theo của hướng dẫn này.

* Tóan tử logic

Toán tử logic được sử dụng để xác định logic giữa các biến hoặc giá trị.

Cho rằng x = 6và y = 3, bảng dưới đây giải thích các toán tử logic:

|  |  |  |
| --- | --- | --- |
| Operator | Description | Example |
| && | and | (x < 10 && y > 1) is true |
| || | or | (x == 5 || y == 5) is false |
| ! | not | !(x == y) is true |

* Toán tử có điều kiện (Ternary)

JavaScript cũng chứa một toán tử có điều kiện gán giá trị cho một biến dựa trên một số điều kiện.

Cú pháp

variablename = (condition) ? value1:value2

Thí dụ:

var voteable = (age < 18) ? "Too young":"Old enough";

Nếu tuổi biến là giá trị dưới 18, giá trị của biến có thể bỏ phiếu sẽ là "Quá trẻ", nếu không, giá trị có thể bỏ phiếu sẽ là "Đủ cũ".

* So sánh các loại khác nhau

So sánh dữ liệu của các loại khác nhau có thể cho kết quả bất ngờ.Khi so sánh một chuỗi với một số, JavaScript sẽ chuyển đổi chuỗi thành một số khi thực hiện so sánh. Một chuỗi rỗng chuyển đổi thành 0. Một chuỗi không phải là số chuyển đổi thành NaNluôn luôn false.

|  |  |
| --- | --- |
| Case | Value |
| 2 < 12 | true |
| 2 < "12" | true |
| 2 < "John" | false |
| 2 > "John" | false |
| 2 == "John" | false |
| "2" < "12" | false |
| "2" > "12" | true |
| "2" == "12" | false |

Khi so sánh hai chuỗi, "2" sẽ lớn hơn "12", bởi vì (theo bảng chữ cái) 1 nhỏ hơn 2.

Để bảo đảm một kết quả phù hợp, các biến phải được chuyển đổi thành loại thích hợp trước khi so sánh

#### JS Conditions

* Câu điều kiện

Rất thường xuyên khi bạn viết mã, bạn muốn thực hiện các hành động khác nhau cho các quyết định khác nhau.

Bạn có thể sử dụng các câu lệnh có điều kiện trong mã của bạn để làm điều này.

Trong JavaScript, chúng tôi có các câu điều kiện sau:

Sử dụng ifđể chỉ định một khối mã sẽ được thực thi, nếu một điều kiện được chỉ định là đúng

Sử dụng elseđể chỉ định một khối mã sẽ được thực thi, nếu điều kiện tương tự là sai

Sử dụng else ifđể chỉ định một điều kiện mới để kiểm tra, nếu điều kiện đầu tiên là sai

Sử dụng switchđể chỉ định nhiều khối mã thay thế sẽ được thực thi

Các switchtuyên bố được mô tả trong chương kế tiếp.

* Câu lệnh if

Sử dụng ifcâu lệnh để chỉ định một khối mã JavaScript sẽ được thực thi nếu một điều kiện là đúng.

Cú pháp

if (condition) {  
  //  block of code to be executed if the condition is true  
}

Lưu ý rằng đó iflà chữ thường. Các chữ cái viết hoa (Nếu hoặc IF) sẽ tạo ra lỗi JavaScript.

Thí dụ

Thực hiện lời chào "Chúc một ngày tốt lành" nếu giờ ít hơn 18:00:

if (hour < 18) {  
  greeting = "Good day";  
}

* Tuyên bố khác

Sử dụng elsecâu lệnh để chỉ định một khối mã sẽ được thực thi nếu điều kiện sai.

if (condition) {  
  //  block of code to be executed if the condition is true  
} else {  
  //  block of code to be executed if the condition is false  
}

Thí dụ:

Nếu giờ dưới 18, hãy tạo một lời chào "Chúc một ngày tốt lành", nếu không thì "Chào buổi tối":

if (hour < 18) {  
  greeting = "Good day";  
} else {  
  greeting = "Good evening";  
}

Kết quả của lời chào sẽ là:

Good evening

* Mặt khác nếu Statement

Sử dụng else ifcâu lệnh để chỉ định một điều kiện mới nếu điều kiện đầu tiên là sai.

Cú pháp

if (condition1) {  
  //  block of code to be executed if condition1 is true  
} else if (condition2) {  
  //  block of code to be executed if the condition1 is false and condition2 is true  
} else {  
  //  block of code to be executed if the condition1 is false and condition2 is false  
}

Thí dụ

Nếu thời gian ít hơn 10:00, hãy tạo một lời chào "Chào buổi sáng", nếu không, nhưng thời gian ít hơn 20:00, hãy tạo một lời chào "Chúc một ngày tốt lành", nếu không thì là "Chào buổi tối":

if (time < 10) {  
  greeting = "Good morning";  
} else if (time < 20) {  
  greeting = "Good day";  
} else {  
  greeting = "Good evening";  
}

Kết quả của lời chào sẽ là:

Good evening

#### JS Switch

Các switchtuyên bố được sử dụng để thực hiện các hành động khác nhau dựa trên các điều kiện khác nhau.

* Tuyên bố chuyển đổi JavaScript

Sử dụng switchcâu lệnh để chọn một trong nhiều khối mã sẽ được thực thi.

Cú pháp

switch(expression) {  
  case x:  
    *// code block*    break;  
  case y:  
    *// code block*    break;  
  default:  
    // code block  
}

Đây là cách nó hoạt động:

Biểu thức chuyển đổi được đánh giá một lần.

Giá trị của biểu thức được so sánh với các giá trị của từng trường hợp.

Nếu có sự trùng khớp, khối mã liên quan được thực thi.

Thí dụ:

Các getDay()phương thức trả về các ngày trong tuần như một số giữa 0 và 6.

(Chủ nhật = 0, Thứ hai = 1, Thứ ba = 2 ..)

Ví dụ này sử dụng số ngày trong tuần để tính tên ngày trong tuần:

switch (new Date().getDay()) {  
  case 0:  
    day = "Sunday";  
    break;  
  case 1:  
    day = "Monday";  
    break;  
  case 2:  
     day = "Tuesday";  
    break;  
  case 3:  
    day = "Wednesday";  
    break;  
  case 4:  
    day = "Thursday";  
    break;  
  case 5:  
    day = "Friday";  
    break;  
  case 6:  
    day = "Saturday";  
}

Kết quả của ngày sẽ là:

Friday

* Từ khóa phá vỡ

Khi JavaScript đạt đến một break từ khóa, nó sẽ thoát ra khỏi khối chuyển đổi.

Điều này sẽ dừng việc thực hiện bên trong khối.

Không cần thiết phải phá vỡ trường hợp cuối cùng trong một khối chuyển đổi. Các khối phá vỡ (kết thúc) ở đó anyway.

Lưu ý: Nếu bạn bỏ qua câu lệnh break, trường hợp tiếp theo sẽ được thực thi ngay cả khi đánh giá không khớp với trường hợp.

* Từ khóa mặc định

Các defaulttừ khóa chỉ định mã để chạy nếu không có trận đấu trường hợp:

Thí dụ

Các getDay()phương thức trả về các ngày trong tuần như một số giữa 0 và 6.

Nếu hôm nay không phải là thứ bảy (6) cũng không phải chủ nhật (0), hãy viết một tin nhắn mặc định:

switch (new Date().getDay()) {  
  case 6:  
    text = "Today is Saturday";  
    break;  
  case 0:  
    text = "Today is Sunday";  
    break;  
  default:  
    text = "Looking forward to the Weekend";  
}

Kết quả của văn bản sẽ là:

Looking forward to the Weekend

Các defaulttrường hợp không phải là trường hợp cuối cùng trong một khối chuyển đổi:

Thí dụ:

switch (new Date().getDay()) {  
  default:  
    text = "Looking forward to the Weekend";  
    break;  
  case 6:  
    text = "Today is Saturday";  
    break;  
  case 0:  
    text = "Today is Sunday";  
}

Nếu defaultkhông phải là trường hợp cuối cùng trong khối chuyển đổi, hãy nhớ kết thúc trường hợp mặc định bằng dấu ngắt.

* Khối mã chung

Đôi khi bạn sẽ muốn các trường hợp chuyển đổi khác nhau để sử dụng cùng một mã.

Trong ví dụ này, trường hợp 4 và 5 chia sẻ cùng một khối mã và 0 và 6 chia sẻ một khối mã khác:

Thí dụ

switch (new Date().getDay()) {  
  case 4:  
  case 5:  
    text = "Soon it is Weekend";  
    break;  
  case 0:  
  case 6:  
    text = "It is Weekend";  
    break;  
  default:  
    text = "Looking forward to the Weekend";  
}

* Chi tiết chuyển đổi

Nếu nhiều trường hợp khớp với một giá trị trường hợp, trường hợp đầu tiên được chọn.

Nếu không tìm thấy trường hợp khớp nào, chương trình sẽ tiếp tục nhãn mặc định .

Nếu không tìm thấy nhãn mặc định, chương trình sẽ tiếp tục (các) câu lệnh sau khi chuyển đổi

* So sánh nghiêm ngặt

Trường hợp chuyển đổi sử dụng so sánh nghiêm ngặt (===).

Các giá trị phải cùng loại để phù hợp.

Một so sánh nghiêm ngặt chỉ có thể đúng nếu các toán hạng cùng loại.

Trong ví dụ này sẽ không có kết quả khớp nào cho x:

Thí dụ:

var x = "0";  
switch (x) {  
  case 0:  
    text = "Off";  
    break;  
  case 1:  
    text = "On";  
    break;  
  default:  
    text = "No value found";  
}

#### JS Loop For

Vòng lặp có thể thực thi một khối mã nhiều lần.

* Vòng lặp JavaScript

Vòng lặp rất tiện dụng, nếu bạn muốn chạy cùng một mã nhiều lần, mỗi lần có một giá trị khác nhau.

Thường thì đây là trường hợp khi làm việc với mảng:

Thay vì viết:

text += cars[0] + "<br>";  
text += cars[1] + "<br>";  
text += cars[2] + "<br>";  
text += cars[3] + "<br>";  
text += cars[4] + "<br>";  
text += cars[5] + "<br>";

Bạn có thể viết:

var i;  
for (i = 0; i < cars.length; i++) {  
  text += cars[i] + "<br>";  
}

* Các loại vòng khác nhau

JavaScript hỗ trợ các loại vòng lặp khác nhau:

for - vòng lặp thông qua một khối mã nhiều lần

for/in - vòng lặp thông qua các thuộc tính của một đối tượng

for/of - vòng lặp thông qua các giá trị của một đối tượng lặp

while - các vòng lặp thông qua một khối mã trong khi một điều kiện được chỉ định là đúng

do/while - cũng lặp qua một khối mã trong khi một điều kiện được chỉ định là đúng

* Vòng lặp

Các forvòng lặp có cú pháp như sau:

for (statement 1; statement 2; statement 3) {  
  // code block to be executed  
}

Câu lệnh 1 được thực thi (một lần) trước khi thực thi khối mã.

Câu 2 xác định điều kiện để thực thi khối mã.

Câu lệnh 3 được thực thi (mọi lúc) sau khi khối mã được thực thi.

Thí dụ

for (i = 0; i < 5; i++) {  
  text += "The number is " + i + "<br>";  
}

Từ ví dụ trên, bạn có thể đọc:

Câu lệnh 1 đặt một biến trước khi vòng lặp bắt đầu (var i = 0).

Câu 2 xác định điều kiện để vòng lặp chạy (i phải nhỏ hơn 5).

Câu lệnh 3 tăng một giá trị (i ++) mỗi lần khối mã trong vòng lặp được thực thi.

* Tuyên bố 1

Thông thường bạn sẽ sử dụng câu lệnh 1 để khởi tạo biến được sử dụng trong vòng lặp (i = 0).

Điều này không phải lúc nào cũng đúng, JavaScript không quan tâm. Tuyên bố 1 là tùy chọn.

Bạn có thể bắt đầu nhiều giá trị trong câu lệnh 1 (được phân tách bằng dấu phẩy):

Thí dụ

for (i = 0, len = cars.length, text = ""; i < len; i++) {  
  text += cars[i] + "<br>";  
}

Và bạn có thể bỏ qua câu lệnh 1 (như khi giá trị của bạn được đặt trước khi vòng lặp bắt đầu):

Thí dụ

var i = 2;  
var len = cars.length;  
var text = "";  
for (; i < len; i++) {  
  text += cars[i] + "<br>";  
}

* Tuyên bố 2

Câu lệnh 2 thường được sử dụng để đánh giá điều kiện của biến ban đầu.

Điều này không phải lúc nào cũng đúng, JavaScript không quan tâm. Tuyên bố 2 cũng là tùy chọn.

Nếu câu lệnh 2 trả về true, vòng lặp sẽ bắt đầu lại, nếu nó trả về false, vòng lặp sẽ kết thúc.

Nếu bạn bỏ qua câu lệnh 2, bạn phải cung cấp một dấu ngắt trong vòng lặp. Nếu không thì vòng lặp sẽ không bao giờ kết thúc. Điều này sẽ làm sập trình duyệt của bạn. Đọc về nghỉ trong một chương sau của hướng dẫn này.

* Tuyên bố 3

Câu lệnh 3 thường tăng giá trị của biến ban đầu.

Điều này không phải lúc nào cũng đúng, JavaScript không quan tâm và câu lệnh 3 là tùy chọn.

Tuyên bố 3 có thể làm bất cứ điều gì như tăng âm (i--), tăng dương (i = i + 15) hoặc bất cứ điều gì khác.

Câu lệnh 3 cũng có thể được bỏ qua (như khi bạn tăng các giá trị của mình bên trong vòng lặp):

Thí dụ

var i = 0;  
var len = cars.length;  
for (; i < len; ) {  
  text += cars[i] + "<br>";  
  i++;  
}

* Vòng lặp cho / vào

Câu for/inlệnh JavaScript lặp qua các thuộc tính của một đối tượng:

Thí dụ

var person = {fname:"John", lname:"Doe", age:25};  
var text = "";  
var x;  
for (x in person) {  
  text += person[x];  
}

* Vòng / Cho

Câu for/oflệnh JavaScript lặp qua các giá trị của một đối tượng có thể lặp lại

for/of cho phép bạn lặp qua các cấu trúc dữ liệu có thể lặp lại như Mảng, Chuỗi, Bản đồ, NodeLists, v.v.

Các for/ofvòng lặp có cú pháp như sau:

for (variable of iterable) {  
  // code block to be executed  
}

biến - Với mỗi lần lặp, giá trị của thuộc tính tiếp theo được gán cho biến. Biến có thể được khai báo với const, lethoặc var.

iterable - Một đối tượng có các thuộc tính lặp.

Vòng qua một mảng

Thí dụ

var cars = ['BMW', 'Volvo', 'Mini'];  
var x;  
for (x of cars) {  
  document.write(x + "<br >");  
}

Vòng qua chuỗi

Thí dụ

var txt = 'JavaScript';  
var x;  
for (x of txt) {  
  document.write(x + "<br >");  
}

* Vòng lặp While

Các whilevòng lặp và do/whilevòng lặp sẽ được giải thích trong chương kế tiếp.

#### JS Loop While

Các vòng lặp có thể thực thi một khối mã miễn là một điều kiện được chỉ định là đúng.

* Vòng lặp While

Các whilevòng lặp thông qua một khối mã miễn là một điều kiện được chỉ định là đúng.

Cú pháp

while (condition) {  
  // code block to be executed  
}

Thí dụ:

Trong ví dụ sau, mã trong vòng lặp sẽ chạy, lặp đi lặp lại, miễn là một biến (i) nhỏ hơn 10:

Thí dụ:

while (i < 10) {  
  text += "The number is " + i;  
  i++;  
}

Nếu bạn quên tăng biến được sử dụng trong điều kiện, vòng lặp sẽ không bao giờ kết thúc. Điều này sẽ làm sập trình duyệt của bạn.

* Vòng lặp Do / While

Các do/whilevòng lặp là một biến thể của vòng lặp while. Vòng lặp này sẽ thực thi khối mã một lần, trước khi kiểm tra xem điều kiện có đúng không, sau đó nó sẽ lặp lại vòng lặp miễn là điều kiện đúng.

do {  
  // code block to be executed  
}  
while (condition);

Thí dụ:

Ví dụ dưới đây sử dụng một do/whilevòng lặp. Vòng lặp sẽ luôn được thực thi ít nhất một lần, ngay cả khi điều kiện sai, bởi vì khối mã được thực thi trước khi điều kiện được kiểm tra:

Thí dụ

do {  
  text += "The number is " + i;  
  i++;  
}  
while (i < 10);

* So sánh cho và trong khi

Nếu bạn đã đọc chương trước, về vòng lặp for, bạn sẽ phát hiện ra rằng vòng lặp while giống như vòng lặp for, với câu lệnh 1 và câu lệnh 3 bị bỏ qua.

Vòng lặp trong ví dụ này sử dụng forvòng lặp để thu thập tên xe từ mảng ô tô:

Thí dụ

var cars = ["BMW", "Volvo", "Saab", "Ford"];  
var i = 0;  
var text = "";  
for (;cars[i];) {  
  text += cars[i] + "<br>";  
  i++;  
}

Vòng lặp trong ví dụ này sử dụng whilevòng lặp để thu thập tên xe từ mảng ô tô:

Thí dụ:

var cars = ["BMW", "Volvo", "Saab", "Ford"];  
var i = 0;  
var text = "";  
while (cars[i]) {  
  text += cars[i] + "<br>";  
  i++;  
}

#### JS Break

Câu breaklệnh "nhảy ra" của một vòng lặp.

Câu continuelệnh "nhảy qua" một lần lặp trong vòng lặp.

* The Break Statement

Bạn đã thấy breaktuyên bố được sử dụng trong một chương trước của hướng dẫn này. Nó được sử dụng để "nhảy ra" một switch()tuyên bố.

Câu breaklệnh cũng có thể được sử dụng để nhảy ra khỏi vòng lặp.

Câu breaklệnh phá vỡ vòng lặp và tiếp tục thực thi mã sau vòng lặp (nếu có):

Thí dụ:

for (i = 0; i < 10; i++) {  
  if (i === 3) { break; }  
  text += "The number is " + i + "<br>";  
}

* Tuyên bố tiếp tục

Câu continuelệnh phá vỡ một lần lặp (trong vòng lặp), nếu một điều kiện được chỉ định xảy ra và tiếp tục với lần lặp tiếp theo trong vòng lặp.

Ví dụ này bỏ qua giá trị của 3:

Thí dụ

for (i = 0; i < 10; i++) {  
  if (i === 3) { continue; }  
  text += "The number is " + i + "<br>";  
}

* Nhãn JavaScript

Để gắn nhãn các câu lệnh JavaScript, bạn đặt trước các câu lệnh bằng tên nhãn và dấu hai chấm:

label:  
statements

Các breakvà continuetuyên bố là những điều khoản Javascript duy nhất có thể "nhảy ra khỏi" một khối mã.

Cú pháp:

break labelname;  
continue labelname;

Câu continue lệnh (có hoặc không có tham chiếu nhãn) chỉ có thể được sử dụng để bỏ qua một vòng lặp .

Câu breaklệnh, không có tham chiếu nhãn, chỉ có thể được sử dụng để nhảy ra khỏi vòng lặp hoặc công tắc .

Với tham chiếu nhãn, câu lệnh break có thể được sử dụng để nhảy ra khỏi bất kỳ khối mã nào :

Example

var cars = ["BMW", "Volvo", "Saab", "Ford"];  
list: {  
  text += cars[0] + "<br>";  
  text += cars[1] + "<br>";  
  break list;  
  text += cars[2] + "<br>";  
  text += cars[3] + "<br>";  
}

#### JS Type Conversion

Số () chuyển đổi thành Số, Chuỗi () chuyển đổi thành Chuỗi, Boolean () chuyển đổi thành Boolean.

* Các kiểu dữ liệu JavaScript

Trong JavaScript có 5 loại dữ liệu khác nhau có thể chứa các giá trị:

* string
* number
* boolean
* object
* function

Có 6 loại đối tượng:

* Object
* Date
* Array
* String
* Number
* Boolean

Và 2 loại dữ liệu không thể chứa giá trị:

* null
* undefined
* Toán tử typeof

Bạn có thể sử dãy quan sát:

* Kiểu dữ liệu của NaN là số
* Kiểu dữ liệu của một mảng là đối tượng
* Kiểu dữ liệu của một ngày là đối tượng
* Kiểu dữ liệu của null là đối tượng
* Kiểu dữ liệu của một biến không xác định là không xác định \*
* Kiểu dữ liệu của một biến chưa được gán giá trị cũng không được xác địnhụng typeoftoán tử để tìm kiểu dữ liệu của biến JavaScript.
* Kiểu dữ liệu của typeof

Các typeofnhà điều hành không phải là một biến. Nó là một nhà điều hành. Toán tử (+ - \* /) không có bất kỳ kiểu dữ liệu nào.

Nhưng, typeoftoán tử luôn trả về một chuỗi (chứa loại toán hạng).

Tài sản xây dựng

Các constructortài sản trả về hàm xây dựng cho tất cả các biến JavaScript

* Chuyển đổi loại JavaScript

Các biến JavaScript có thể được chuyển đổi thành một biến mới và một kiểu dữ liệu khác:

Bằng cách sử dụng chức năng JavaScript

Tự động bằng chính JavaScript

Chuyển đổi số thành chuỗi

Phương thức toàn cầu String()có thể chuyển đổi số thành chuỗi.

Nó có thể được sử dụng trên bất kỳ loại số, chữ, biến hoặc biểu thức:

Thí dụ:

String(x)         // returns a string from a number variable x  
String(123)       // returns a string from a number literal 123  
String(100 + 23)  // returns a string from a number from an expression

* Chuyển đổi Booleans thành Chuỗi

Phương thức toàn cầu String()có thể chuyển đổi booleans thành chuỗi.

String(false)      // returns "false"  
String(true)       // returns "true"

Phương thức Boolean toString()làm tương tự.

false.toString()   // returns "false"  
true.toString()    // returns "true"

Chuyển đổi ngày thành chuỗi

Phương thức toàn cầu String()có thể chuyển đổi ngày thành chuỗi.

String(Date())  // returns "Thu Jul 17 2014 15:38:19 GMT+0200 (W. Europe Daylight Time)"

Phương thức Date toString()thực hiện tương tự.

Date().toString()  // returns "Thu Jul 17 2014 15:38:19 GMT+0200 (W. Europe Daylight Time)"

* Toán tử Unary +

Các nhà điều hành + unary thể được sử dụng để chuyển đổi một biến với một số

var y = "5";      // y is a string  
var x = + y;      // x is a number

* Chuyển đổi Booleans thành số

Phương pháp toàn cầu Number()cũng có thể chuyển đổi booleans thành số.

Number(false)     // returns 0  
Number(true)      // returns 1

* Chuyển đổi ngày thành số

Phương pháp toàn cầu Number()có thể được sử dụng để chuyển đổi ngày thành số.

d = new Date();  
Number(d)          // returns 1404568027739

Phương pháp ngày getTime() làm tương tự.

d = new Date();  
d.getTime()        // returns 1404568027739

* Chuyển đổi loại tự động

Khi JavaScript cố gắng hoạt động trên loại dữ liệu "sai", nó sẽ cố gắng chuyển đổi giá trị thành loại "đúng"

document.getElementById("demo").innerHTML = myVar;  
// if myVar = {name:"Fjohn"}  // toString converts to "[object Object]"  
// if myVar = [1,2,3,4]       // toString converts to "1,2,3,4"  
// if myVar = new Date()      // toString converts to "Fri Jul 18 2014 09:08:55 GMT+0200"

Số và booleans cũng được chuyển đổi, nhưng điều này không rõ ràng lắm:

// if myVar = 123             // toString converts to "123"  
// if myVar = true            // toString converts to "true"  
// if myVar = false           // toString converts to "false"

#### JS Bitwise

* Toán tử bitwise JavaScript

|  |  |  |
| --- | --- | --- |
| Operator | Name | Description |
| & | AND | Sets each bit to 1 if both bits are 1 |
| | | OR | Sets each bit to 1 if one of two bits is 1 |
| ^ | XOR | Sets each bit to 1 if only one of two bits is 1 |
| ~ | NOT | Inverts all the bits |
| << | Zero fill left shift | Shifts left by pushing zeros in from the right and let the leftmost bits fall off |
| >> | Signed right shift | Shifts right by pushing copies of the leftmost bit in from the left, and let the rightmost bits fall off |
| >>> | Zero fill right shift | Shifts right by pushing zeros in from the left, and let the rightmost bits fall off |

#### JS RegExp

Biểu thức chính quy là một chuỗi các ký tự tạo thành một mẫu tìm kiếm.

Mẫu tìm kiếm có thể được sử dụng cho các hoạt động tìm kiếm văn bản và thay thế văn bản.

* Biểu thức chính quy là gì?

Biểu thức chính quy là một chuỗi các ký tự tạo thành một mẫu tìm kiếm .

Khi bạn tìm kiếm dữ liệu trong một văn bản, bạn có thể sử dụng mẫu tìm kiếm này để mô tả những gì bạn đang tìm kiếm.

Một biểu thức chính quy có thể là một ký tự đơn hoặc một mẫu phức tạp hơn.

Biểu thức thông thường có thể được sử dụng để thực hiện tất cả các loại hoạt động tìm kiếm văn bản và thay thế văn bản .

* Cú pháp

var patt = /w3schools/i;

* Sử dụng các phương thức chuỗi

Trong JavaScript, các biểu thức chính quy thường được sử dụng với hai phương thức chuỗi : search()và replace().

Các search()phương pháp sử dụng một biểu thức để tìm kiếm một trận đấu, và trả về vị trí của trận đấu.

Các replace()phương thức trả về một chuỗi biến đổi nơi mô hình được thay thế

* Sử dụng chuỗi tìm kiếm () với một chuỗi

Thí dụ

Sử dụng một chuỗi để thực hiện tìm kiếm "W3schools" trong một chuỗi:

var str = "Visit W3Schools!";  
var n = str.search("W3Schools");

* Using String search() With a Regular Expression

Thí dụ: sử dụng regularr expression để phân biệt chữ hoa chữ thường.

var str = "Visit W3Schools";  
var n = str.search(/w3schools/i);

Kết quả trong n sẽ là:

6

* Sử dụng replace() bằng 1 chuỗi

Thay thế 1 giá trị xác định với một giá trị trong chuỗi

var str = "Visit Microsoft!";  
var res = str.replace("Microsoft", "W3Schools");

* Sử dụng replace() bằng một biểu thức thông thường.

Sử dụng biểu thức chính quy không phân biệt chữ hoa chữ thường để thay thế Microsoft bằng W3Schools trong một chuỗi:

var str = "Visit Microsoft!";  
var res = str.replace(/microsoft/i, "W3Schools");

Kết quả trong độ phân giải sẽ là:

Visit W3Schools!

* Sử dụng đối tượng RegExp

Trong JavaScript, đối tượng RegExp là một đối tượng biểu thức chính quy với các thuộc tính và phương thức được xác định trước.

* Sử dụng thử nghiệm ()

Các test()phương pháp là một phương pháp biểu hiện RegExp.

Nó tìm kiếm một chuỗi cho một mẫu và trả về giá trị đúng hoặc sai, tùy thuộc vào kết quả.

Ví dụ sau đây tìm kiếm một chuỗi cho ký tự "e":

var patt = /e/;  
patt.test("The best things in life are free!");

Vì có một "e" trong chuỗi, đầu ra của mã ở trên sẽ là:

True

* Sử dụng exec ()

Các exec()phương pháp là một phương pháp biểu hiện RegExp.

Nó tìm kiếm một chuỗi cho một mẫu xác định và trả về văn bản tìm thấy dưới dạng một đối tượng.

Nếu không tìm thấy kết quả khớp, nó sẽ trả về một đối tượng rỗng (null) .

Ví dụ sau đây tìm kiếm một chuỗi cho ký tự "e":

/e/.exec("The best things in life are free!");

#### JS Errors

Câu try lệnh cho phép bạn kiểm tra một khối mã lỗi.

Câu catch lệnh cho phép bạn xử lý lỗi.

Câu throw lệnh cho phép bạn tạo các lỗi tùy chỉnh.

Câu finally lệnh cho phép bạn thực thi mã, sau khi thử và bắt, bất kể kết quả.

* JavaScript try và catch

Câu try lệnh cho phép bạn xác định một khối mã được kiểm tra lỗi trong khi nó đang được thực thi.Câu catch lệnh cho phép bạn xác định một khối mã sẽ được thực thi, nếu xảy ra lỗi trong khối thử.Các câu lệnh JavaScript try và catch đi theo cặp’

try {  
  Block of code to try  
}  
catch(err) {  
  Block of code to handle errors  
}

* Lỗi throw JavaScript

Khi xảy ra lỗi, JavaScript thường sẽ dừng và tạo thông báo lỗi.

Thuật ngữ kỹ thuật cho việc này là: JavaScript sẽ đưa ra một ngoại lệ (đưa ra lỗi) .

JavaScript thực sự sẽ tạo một đối tượng Error với hai thuộc tính: tên và thông báo .

* Throws errors

Câu throw lệnh cho phép bạn tạo một lỗi tùy chỉnh.

Về mặt kỹ thuật bạn có thể ném một ngoại lệ (ném lỗi) .

Ngoại lệ có thể là JavaScript String, a Number, a Boolean hoặc Object:

throw "Too big"; // throw a text

throw 500; // throw a number

Nếu bạn sử dụng throw cùng với try và catch, bạn có thể kiểm soát luồng chương trình và tạo thông báo lỗi tùy chỉnh

* Xác thực HTML

Mã ở trên chỉ là một ví dụ.

Các trình duyệt hiện đại thường sẽ sử dụng kết hợp JavaScript và xác thực HTML tích hợp, sử dụng các quy tắc xác thực được xác định trước được xác định trong các thuộc tính HTML:

<input id="demo" type="number" min="5" max="10" step="1">

Bạn có thể đọc thêm về xác nhận mẫu trong một chương sau của hướng dẫn này.

* Finally statements

Câu finallylệnh cho phép bạn thực thi mã, sau khi thử và bắt, bất kể kết quả:

Cú pháp

try {  
  Block of code to try  
}  
catch(err) {  
  Block of code to handle errors  
}  
finally {  
  Block of code to be executed regardless of the try / catch result  
}

* Đối tượng lỗi

JavaScript có một đối tượng lỗi tích hợp cung cấp thông tin lỗi khi xảy ra lỗi.

Đối tượng lỗi cung cấp hai thuộc tính hữu ích: tên và thông báo.

* Giá trị tên lỗi

|  |  |
| --- | --- |
| Tên lỗi | Sự miêu tả |
| EvalError | Đã xảy ra lỗi trong hàm eval () |
| Phạm viError | Một số "ngoài phạm vi" đã xảy ra |
| Tham khảoError | Một tài liệu tham khảo bất hợp pháp đã xảy ra |
| Lỗi cú pháp | Đã xảy ra lỗi cú pháp |
| LoạiError | Đã xảy ra lỗi loại |
| ĐÔ THỊ | Đã xảy ra lỗi trong encodeURI () |

* Evalerror

An EvalErrorchỉ ra lỗi trong hàm eval ().Các phiên bản mới hơn của JavaScript không ném EvalError. Sử dụng SyntaxError thay thế.

* Lỗi phạm vi

A RangeErrorđược ném nếu bạn sử dụng một số nằm ngoài phạm vi của các giá trị pháp lý.

Ví dụ: Bạn không thể đặt số chữ số có nghĩa của một số thành 500.

* Lỗi tham chiếu

A ReferenceErrorđược ném nếu bạn sử dụng (tham chiếu) một biến chưa được khai báo

* Lỗi cú pháp

A SyntaxErrorbị ném nếu bạn cố gắng đánh giá mã với lỗi cú pháp.

* Lỗi loại

A TypeErrorđược ném nếu bạn sử dụng một giá trị nằm ngoài phạm vi của các loại dự kiến

* Lỗi URI (Mã định danh tài nguyên đồng nhất)

A URIErrorbị ném nếu bạn sử dụng các ký tự không hợp lệ trong hàm URI

* Thuộc tính đối tượng lỗi không chuẩn

Mozilla và Microsoft định nghĩa một số thuộc tính đối tượng lỗi không chuẩn:

fileName (Mozilla)

lineNumber (Mozilla)

cộtNumber (Mozilla)

stack (Mozilla)

mô tả (Microsoft)

số (Microsoft)

Không sử dụng các tài sản này trong các trang web công cộng. Chúng sẽ không hoạt động trong tất cả các trình duyệt.

* Tham chiếu lỗi hoàn toàn

#### JS Scope

Phạm vi xác định khả năng truy cập (khả năng hiển thị) của các biến.

* Phạm vi chức năng JavaScript

Trong JavaScript có hai loại phạm vi:

Phạm vi địa phương

Phạm vi toàn cầu

JavaScript có phạm vi chức năng: Mỗi chức năng tạo ra một phạm vi mới.

Phạm vi xác định khả năng truy cập (khả năng hiển thị) của các biến này.

Các biến được định nghĩa bên trong hàm không thể truy cập (hiển thị) từ bên ngoài hàm.

* Biến JavaScript cục bộ

Các biến được khai báo trong hàm JavaScript, trở thành LỘC cho hàm.

Các biến cục bộ có phạm vi Hàm : Chúng chỉ có thể được truy cập từ bên trong hàm.

Thí dụ

// code here can NOT use carName  
function myFunction() {  
  var carName = "Volvo";  
// code here CAN use carName}

Vì các biến cục bộ chỉ được nhận dạng bên trong các hàm của chúng, các biến có cùng tên có thể được sử dụng trong các hàm khác nhau.

Biến cục bộ được tạo khi hàm bắt đầu và bị xóa khi hàm hoàn thành.

* Biến JavaScript toàn cầu

Một biến được khai báo bên ngoài một hàm, trở thành TOÀN CẦU .

Một biến toàn cục có phạm vi toàn cầu : Tất cả các tập lệnh và chức năng trên một

trang web có thể truy cập nó.

Thí dụ

var carName = "Volvo";  
// code here can use carName  
function myFunction() {  
 // code here can also use carName  
}

* Biến JavaScript

Trong JavaScript, các đối tượng và hàm cũng là các biến.Phạm vi xác định khả năng truy cập của các biến, đối tượng và hàm từ các phần khác nhau của mã.

* Tự động toàn cầu

Nếu bạn gán một giá trị cho một biến chưa được khai báo, nó sẽ tự động trở thành biến GLOBAL .

Ví dụ mã này sẽ khai báo một biến toàn cục carName, ngay cả khi giá trị được gán bên trong một hàm.

Thí dụ

myFunction();  
// code here can use carName  
function myFunction() {  
  carName = "Volvo";  
}

* Chế độ nghiêm ngặt

Tất cả các trình duyệt hiện đại đều hỗ trợ chạy JavaScript trong "Chế độ nghiêm ngặt".

Bạn sẽ tìm hiểu thêm về cách sử dụng chế độ nghiêm ngặt trong chương sau của hướng dẫn này.

Biến toàn cục không được tạo tự động trong "Chế độ nghiêm ngặt".

* Biến toàn cầu trong HTML

Với JavaScript, phạm vi toàn cầu là môi trường JavaScript hoàn chỉnh.

Trong HTML, phạm vi toàn cầu là đối tượng cửa sổ. Tất cả các biến toàn cục thuộc về đối tượng cửa sổ.

Thí dụ

var carName = "Volvo";  
// code here can use window.carName

* Cảnh báo

KHÔNG tạo các biến toàn cục trừ khi bạn có ý định.

Các biến toàn cục (hoặc hàm) của bạn có thể ghi đè lên các biến cửa sổ (hoặc hàm).

Bất kỳ chức năng nào, bao gồm cả đối tượng cửa sổ, có thể ghi đè các biến và hàm toàn cục của bạn.

Thời gian tồn tại của các biến JavaScript

Thời gian tồn tại của một biến JavaScript bắt đầu khi nó được khai báo.

Biến cục bộ bị xóa khi chức năng hoàn thành.

Trong trình duyệt web, các biến toàn cục sẽ bị xóa khi bạn đóng cửa sổ trình duyệt (hoặc tab), nhưng vẫn có sẵn cho các trang mới được tải vào cùng một cửa sổ.

* Đối số chức năng

Các đối số hàm (tham số) hoạt động như các biến cục bộ bên trong các hàm

#### JS Hoisting

Tời là hành vi mặc định của JavaScript về việc di chuyển các khai báo lên đầu.

* JavaScript Declarations are Hoisted

Trong JavaScript, một biến có thể được khai báo sau khi nó được sử dụng.

Nói cách khác; một biến có thể được sử dụng trước khi nó được khai báo

Để hiểu điều này, bạn phải hiểu thuật ngữ "cẩu".

Hoisted là hành vi mặc định của JavaScript là di chuyển tất cả các khai báo lên trên cùng của phạm vi hiện tại (lên đầu tập lệnh hiện tại hoặc hàm hiện tại)

* Từ khóa let và const

Các biến và hằng được khai báo có lethoặc constkhông được nâng lên!

Đọc thêm về let và const trong JS Let / Const

* Khởi tạo JavaScript không được nâng lên

JavaScript chỉ nâng các khai báo, không khởi tạo.

* Khai báo biến của bạn ở đầu!

Tời là (đối với nhiều nhà phát triển) một hành vi chưa biết hoặc bị bỏ qua của JavaScript.

Nếu nhà phát triển không hiểu về cẩu, các chương trình có thể chứa lỗi (lỗi).

Để tránh lỗi, luôn luôn khai báo tất cả các biến ở đầu mọi phạm vi.

Vì đây là cách JavaScript diễn giải mã, nên nó luôn là một quy tắc tốt.

#### JS Strict Mode

"use strict";Xác định rằng mã JavaScript phải được thực thi trong "chế độ nghiêm ngặt".

* Chỉ thị "sử dụng nghiêm ngặt"Lệnh "use strict"này là mới trong ECMAScript phiên bản 5.

Đây không phải là một tuyên bố, mà là một biểu thức nghĩa đen, bị bỏ qua bởi các phiên bản JavaScript trước đó.

Mục đích của "use strict"việc chỉ ra rằng mã phải được thực thi trong "chế độ nghiêm ngặt".

Với chế độ nghiêm ngặt, ví dụ, bạn không thể sử dụng các biến không được khai báo.

Tất cả các trình duyệt hiện đại đều hỗ trợ "sử dụng nghiêm ngặt" ngoại trừ Internet Explorer 9 trở xuống

* Khai báo chế độ nghiêm ngặt

Chế độ nghiêm ngặt được khai báo bằng cách thêm "sử dụng nghiêm ngặt"; đến đầu một kịch bản hoặc một chức năng.

Được khai báo khi bắt đầu tập lệnh, nó có phạm vi toàn cục (tất cả mã trong tập lệnh sẽ thực thi ở chế độ nghiêm ngặt)

Việc "sử dụng nghiêm ngặt"; Cú pháp

Cú pháp, để khai báo chế độ nghiêm ngặt, được thiết kế để tương thích với các phiên bản JavaScript cũ hơn.

Biên dịch một chữ số (4 + 5;) hoặc một chuỗi ký tự ("John Doe";) trong một chương trình JavaScript không có tác dụng phụ. Nó chỉ đơn giản biên dịch thành một biến không tồn tại và chết.

Vì vậy, "use strict";chỉ quan trọng đối với trình biên dịch mới "hiểu" ý nghĩa của nó.

* Tại sao chế độ nghiêm ngặt?

Chế độ nghiêm ngặt giúp viết JavaScript "an toàn" dễ dàng hơn.

Thay đổi chế độ nghiêm ngặt trước đây đã chấp nhận "cú pháp xấu" thành lỗi thực.

Ví dụ, trong JavaScript thông thường, việc nhập sai tên biến sẽ tạo ra một biến toàn cục mới. Trong chế độ nghiêm ngặt, điều này sẽ gây ra lỗi, khiến nó không thể vô tình tạo ra một biến toàn cục.

Trong JavaScript thông thường, nhà phát triển sẽ không nhận được bất kỳ phản hồi lỗi nào khi gán giá trị cho các thuộc tính không thể ghi.

Trong chế độ nghiêm ngặt, bất kỳ sự gán nào cho một thuộc tính không thể ghi, thuộc tính chỉ getter, thuộc tính không tồn tại, biến không tồn tại hoặc đối tượng không tồn tại sẽ gây ra lỗi.

* Không được phép ở Chế độ nghiêm ngặt

Sử dụng một biến, mà không khai báo nó, không được phép

Sử dụng một đối tượng, mà không khai báo nó, không được phép

Xóa một biến (hoặc đối tượng) không được phép

Xóa một chức năng là không được phép

Sao chép tên tham số không được phép

Chữ số bát phân không được phép

Nhân vật thoát bát không được phép

Viết cho một tài sản chỉ đọc không được phép

Viết cho một tài sản chỉ nhận được không được phép

Xóa một tài sản không thể xóa được không được phép

Từ eval không thể được sử dụng như một biến:

Từ arguments không thể được sử dụng như một biến

Các with tuyên bố không được phép

Các this từ khóa trong các chức năng hoạt động khác nhau trong chế độ nghiêm ngặt.

Các this từ khóa liên quan đến các đối tượng đó gọi hàm.

Nếu đối tượng không được chỉ định, các chức năng ở chế độ nghiêm ngặt sẽ trả về undefinedvà các chức năng ở chế độ bình thường sẽ trả về đối tượng toàn cầu (cửa sổ)

* Bằng chứng tương lai!

Từ khóa dành riêng cho các phiên bản JavaScript trong tương lai KHÔNG thể được sử dụng làm tên biến trong chế độ nghiêm ngặt.

Đó là:

thực hiện

giao diện

để cho

gói

riêng tư

được bảo vệ

công cộng

tĩnh

năng suất

#### JS this Keyword

var **person** = {  
  firstName: "John",  
  lastName : "Doe",  
  id       : 5566,  
  fullName : function() {  
    return **this**.firstName + " " + **this**.lastName;  
  }  
};

* Đây là cái gì

this Từ khóa JavaScript đề cập đến đối tượng mà nó thuộc về.

Nó có các giá trị khác nhau tùy thuộc vào nơi nó được sử dụng:

Trong một phương thức, thisđề cập đến đối tượng chủ sở hữu .

Một mình, thisđề cập đến các đối tượng toàn cầu .

Trong một chức năng, thisđề cập đến các đối tượng toàn cầu .

Trong một chức năng, trong chế độ nghiêm ngặt, thislà undefined.

Trong một sự kiện, thisđề cập đến yếu tố đã nhận sự kiện.

Các phương thức như call(), và apply()có thể tham chiếu thisđến bất kỳ đối tượng nào .

* THIS trong một Phương thức

Trong một phương thức đối tượng, thisđề cập đến " chủ sở hữu " của phương thức.

Trong ví dụ trên đầu trang này, thisđề cập đến đối tượng người .

Các người đối tượng là chủ sở hữu của FullName phương pháp.

* THIS ALONE

Khi được sử dụng một mình, chủ sở hữu là đối tượng Toàn cầu, vì vậy this đề cập đến đối tượng Toàn cầu.

Trong cửa sổ trình duyệt, đối tượng Toàn cầu là [object Window]

Trong chế độ nghiêm ngặt , khi được sử dụng một mình, thiscũng đề cập đến đối tượng Toàn cầu [object Window]:

var x = this;

* THIS trong Hàm (Mặc định)

Trong hàm JavaScript, chủ sở hữu của hàm là ràng buộc mặc định cho this.

Vì vậy, trong một chức năng, thisđề cập đến đối tượng Toàn cầu [object Window].

* THIS trong Hàm (Nghiêm)

Chế độ nghiêm ngặt JavaScript không cho phép ràng buộc mặc định.

Vì vậy, khi được sử dụng trong một chức năng, trong chế độ nghiêm ngặt, thislà undefined.

"use strict";  
function myFunction() {  
  return this;  
}

* THIS trong Event Handlers

Trong các trình xử lý sự kiện HTML, thisđề cập đến phần tử HTML đã nhận sự kiện

<button onclick="this.style.display='none'">  
  Click to Remove Me!  
</button>

* Đối tượng Phương pháp ràng buộc

Trong các ví dụ này, thislà đối tượng person (Đối tượng person là "chủ sở hữu" của hàm)

Nói cách khác: this.firstName nghĩa là firstName tài sản của này (người) đối tượng

var **person** = {  
  firstName  : "John",  
  lastName   : "Doe",  
  id         : 5566,  
  myFunction : function() {  
    return **this**;  
  }  
};

var **person** = {  
  firstName: "John",  
  lastName : "Doe",  
  id       : 5566,  
  fullName : function() {  
    return **this**.firstName + " " + **this**.lastName;  
  }  
};

* Chức năng ràng buộc ràng buộc

Các phương thức call()và apply()là các phương thức JavaScript được xác định trước.

Cả hai đều có thể được sử dụng để gọi một phương thức đối tượng với một đối tượng khác làm đối số.

var person1 = {  
  fullName: function() {  
    return this.firstName + " " + this.lastName;  
  }  
}  
var person2 = {  
  firstName:"John",  
  lastName: "Doe",  
}  
person1.fullName.call(person2);  // Will return "John Doe"

#### JS Let

* Bản thảo năm 2015

ES2015 đã giới thiệu hai từ khóa JavaScript mới quan trọng: letvà const.

Hai từ khóa này cung cấp các biến Phạm vi khối (và hằng số) trong JavaScript.

Trước ES2015, JavaScript chỉ có hai loại phạm vi: Phạm vi toàn cầu và Phạm vi chức năng .

* Phạm vi toàn cầu

Các biến được khai báo trên toàn cầu (bên ngoài bất kỳ chức năng nào) có Phạm vi toàn cầu .

Biến toàn cục có thể được truy cập từ bất cứ đâu trong chương trình JavaScript.

var carName = "Volvo";  
// code here can use carName  
function myFunction() {

// code here can also use carName  
}

* Phạm vi chức năng

Các biến được khai báo cục bộ (bên trong một hàm) có Phạm vi hàm .

Các biến cục bộ chỉ có thể được truy cập từ bên trong hàm nơi chúng được khai báo.

// code here can NOT use carName  
function myFunction() {  
  var carName = "Volvo";  
  // code here CAN use carName  
}  
// code here can NOT use carName

* Phạm vi khối JavaScript

Các biến được khai báo cục bộ (bên trong một hàm) có Phạm vi hàm .

Các biến cục bộ chỉ có thể được truy cập từ bên trong hàm nơi chúng được khai báo

{  
  var x = 2;  
}  
// x CAN be used here

Các biến được khai báo với vartừ khóa không thể có Phạm vi khối .

Các biến được khai báo bên trong một khối {} có thể được truy cập từ bên ngoài khối.

{  
  let x = 2;  
}  
// x can NOT be used here

Trước ES2015 JavaScript không có Phạm vi chặn .

Các biến được khai báo với lettừ khóa có thể có Phạm vi khối.

Các biến được khai báo bên trong một khối {} không thể được truy cập từ bên ngoài khối

* Biến đổi lại

Khai báo lại một biến bằng cách sử dụng vartừ khóa có thể áp đặt các vấn đề.

Khai báo lại một biến trong một khối cũng sẽ xác định lại biến bên ngoài khối

var x = 10;  
// Here x is 10  
{  
  var x = 2;  
  // Here x is 2  
}  
// Here x is 2

Khai báo lại một biến bằng cách sử dụng lettừ khóa có thể giải quyết vấn đề này.

Khai báo lại một biến trong một khối sẽ không phân phối lại biến bên ngoài khối

var x = 10;  
// Here x is 10  
{  
  let x = 2;  
  // Here x is 2  
}  
// Here x is 10

* Hỗ trợ trình duyệt

Các lettừ khóa không được hỗ trợ đầy đủ trong Internet Explorer 11 hoặc sớm hơn.

Bảng sau xác định các phiên bản trình duyệt đầu tiên có hỗ trợ đầy đủ cho let từ khóa

* Phạm vi vòng lặp

Sử dụng var trong một vòng lặp

var i = 5;  
for (var i = 0; i < 10; i++) {  
  // some statements  
}  
// Here i is 10

Sử dụng let trong một vòng lặp

let i = 5;  
for (let i = 0; i < 10; i++) {  
  // some statements  
}  
// Here i is 5

Trong ví dụ đầu tiên, bằng cách sử dụng var, biến được khai báo trong vòng lặp sẽ xác định lại biến bên ngoài vòng lặp.

Trong ví dụ thứ hai, sử dụng let, biến được khai báo trong vòng lặp không phân phối lại biến ngoài vòng lặp.

Khi let được sử dụng để khai báo biến i trong một vòng lặp, biến i sẽ chỉ hiển thị trong vòng lặp

* Phạm vi chức năng

Các biến được khai báo với varvà letkhá giống nhau khi được khai báo bên trong một hàm.

Cả hai sẽ có Phạm vi chức năng

function myFunction() {  
  var carName = "Volvo";   // Function Scope  
}

function myFunction() {  
  let carName = "Volvo";   // Function Scope  
}

* Phạm vi toàn cầu

Các biến được khai báo với var và let khá giống nhau khi được khai báo bên ngoài một khối.

Cả hai sẽ có Phạm vi toàn cầu :

var x = 2;       // Global scope

let x = 2;       // Global scope

* Biến toàn cầu trong HTML

Với JavaScript, phạm vi toàn cầu là môi trường JavaScript.

Trong HTML, phạm vi toàn cầu là đối tượng cửa sổ.

Các biến toàn cục được xác định với var từ khóa thuộc về đối tượng cửa sổ

Các biến toàn cục được xác định với lettừ khóa không thuộc về đối tượng cửa sổ

#### JS Const

* Bản thảo năm 2015

ES2015 đã giới thiệu hai từ khóa JavaScript mới quan trọng: letvà const.

Các biến được định nghĩa với các biến constgiống như letcác biến, ngoại trừ chúng không thể được gán lại

* Phạm vi khối

Khai báo một biến const tương tự như let khi nói đến Block Scope .

X được khai báo trong khối, trong ví dụ này, không giống với x được khai báo bên ngoài khối

* Được chỉ định khi khai báo

Các const biến JavaScript phải được gán một giá trị khi chúng được khai báo:

Sai

const PI;

PI = 3.14159265359;

Chính xác

const PI = 3.14159265359;

* Không phải hằng thực

Các từ khóa const là một chút sai lệch.

Nó KHÔNG định nghĩa một giá trị không đổi. Nó định nghĩa một tham chiếu không đổi đến một giá trị.

Do đó, chúng ta không thể thay đổi các giá trị nguyên thủy không đổi, nhưng chúng ta có thể thay đổi các thuộc tính của các đối tượng không đổi

* Giá trị nguyên thủy

Nếu chúng ta gán giá trị nguyên thủy cho hằng số, chúng ta không thể thay đổi giá trị nguyên thủy

* Đối tượng không đổi có thể thay đổi

Bạn có thể thay đổi các thuộc tính của một đối tượng không đổi

Nhưng bạn KHÔNG thể gán lại một đối tượng không đổi

* Mảng liên tục có thể thay đổi

Bạn có thể thay đổi các phần tử của một mảng không đổi

Nhưng bạn KHÔNG thể gán lại một mảng không đổi

* Hỗ trợ trình duyệt

Các consttừ khóa không được hỗ trợ trong Internet Explorer 10 hoặc sớm hơn.

Bảng sau xác định các phiên bản trình duyệt đầu tiên có hỗ trợ đầy đủ cho const từ khóa

* Khai báo

Khai báo lại một biến JavaScript varđược cho phép ở bất cứ đâu trong một chương trình

Không được phép khai báo lại một varbiến có lettrong cùng phạm vi hoặc trong cùng một khối

Không được phép khai báo lại một letbiến có lettrong cùng phạm vi hoặc trong cùng một khối

Không được phép khai báo lại một letbiến có vartrong cùng phạm vi hoặc trong cùng một khối

Việc khai báo lại một biến với let, trong một phạm vi khác hoặc trong một khối khác, được cho phép.

* Tời

Các biến được định nghĩa với varđược nâng lên hàng đầu (nếu bạn không biết Tời là gì, hãy đọc Chương Tời kéo của chúng tôi ).

Bạn có thể sử dụng một biến trước khi nó được khai báo

Các biến được định nghĩa với letkhông được nâng lên trên cùng.

Sử dụng một letbiến trước khi nó được khai báo sẽ dẫn đến a ReferenceError.

Biến nằm trong "vùng chết tạm thời" từ khi bắt đầu khối cho đến khi được khai báo

#### JS Debugging

* Chức năng mũi tên

Các chức năng mũi tên đã được giới thiệu trong ES6.

Các hàm mũi tên cho phép chúng ta viết cú pháp hàm ngắn hơn:

Trước:

hello = function() {

return "Hello World!";

}

Với chức năng mũi tên:

hello = () => {

return "Hello World!";

}

Nó được ngắn hơn! Nếu chức năng chỉ có một tuyên bố, và báo cáo kết quả trả về một giá trị, bạn có thể loại bỏ các dấu ngoặc và các returntừ khóa:

Hàm mũi tên Trả về giá trị theo mặc định:

hello = () => "Hello World!";

Chức năng mũi tên với các tham số:

hello = (val) => "Hello " + val;

Trong thực tế, nếu bạn chỉ có một tham số, bạn cũng có thể bỏ qua dấu ngoặc đơn:

Chức năng mũi tên không có dấu ngoặc đơn:

hello = val => "Hello " + val;

* Thế còn this?

Việc xử lý thiscũng khác nhau trong các chức năng mũi tên so với các chức năng thông thường.

Trong ngắn hạn, với các chức năng mũi tên không có ràng buộc this.

Trong các hàm thông thường, thistừ khóa đại diện cho đối tượng được gọi là hàm, có thể là cửa sổ, tài liệu, nút hoặc bất cứ thứ gì.

Với các hàm mũi tên, thistừ khóa luôn đại diện cho đối tượng đã xác định hàm mũi tên.

Chúng ta hãy xem hai ví dụ để hiểu sự khác biệt.

Cả hai ví dụ đều gọi một phương thức hai lần, lần đầu tiên khi tải trang và một lần nữa khi người dùng nhấp vào nút.

Ví dụ đầu tiên sử dụng hàm thông thường và ví dụ thứ hai sử dụng hàm mũi tên.

Kết quả cho thấy ví dụ đầu tiên trả về hai đối tượng khác nhau (cửa sổ và nút) và ví dụ thứ hai trả về đối tượng cửa sổ hai lần, bởi vì đối tượng cửa sổ là "chủ sở hữu" của hàm.

* Hỗ trợ trình duyệt

Bảng sau xác định các phiên bản trình duyệt đầu tiên có hỗ trợ đầy đủ cho Hàm mũi tên trong JavaScript

#### JS Style Guide

Luôn sử dụng các quy ước mã hóa giống nhau cho tất cả các dự án JavaScript của bạn.

* Các quy ước mã hóa JavaScript
* Quy ước mã hóa là hướng dẫn phong cách cho lập trình . Họ thường bao gồm:
* Quy tắc đặt tên và khai báo cho các biến và chức năng.
* Quy tắc sử dụng khoảng trắng, thụt lề và bình luận.

Thực hành lập trình và nguyên tắc

Công ước mã hóa chất lượng an toàn

* Cải thiện khả năng đọc mã
* Giúp bảo trì mã dễ dàng hơn

Các quy ước mã hóa có thể là tài liệu quy tắc cho các nhóm tuân theo, hoặc chỉ là thực hành mã hóa cá nhân của bạn

* Tên biến

Tất cả các tên bắt đầu bằng một chữ cái .

Ở dưới cùng của trang này, bạn sẽ tìm thấy một cuộc thảo luận rộng hơn về quy tắc đặt tên.

firstName = "John";  
lastName = "Doe";  
price = 19.90;  
tax = 0.20;  
fullPrice = price + (price \* tax);

Không gian xung quanh các nhà khai thác

Luôn đặt khoảng trắng xung quanh toán tử (= + - \* /) và sau dấu phẩy:

Ví dụ:

var x = y + z;  
var values = ["Volvo", "Saab", "Fiat"];

* Code Indentation

Luôn sử dụng 2 khoảng trắng để thụt các khối mã:

Chức năng:

function toCelsius(fahrenheit) {  
  return (5 / 9) \* (fahrenheit - 32);  
}

* Nội quy

Quy tắc chung cho các tuyên bố đơn giản:

Luôn kết thúc một tuyên bố đơn giản bằng dấu chấm phẩy.

Ví dụ:

var values = ["Volvo", "Saab", "Fiat"];  
var person = {  
  firstName: "John",  
  lastName: "Doe",  
  age: 50,  
  eyeColor: "blue"  
};

Quy tắc chung cho các câu lệnh (ghép) phức tạp:

* Đặt khung mở ở cuối dòng đầu tiên.
* Sử dụng một khoảng trắng trước dấu ngoặc mở.
* Đặt khung đóng trên một dòng mới, không có khoảng trắng ở đầu.
* Đừng kết thúc một tuyên bố phức tạp bằng dấu chấm phẩy.

Chức năng:

function toCelsius(fahrenheit) {  
  return (5 / 9) \* (fahrenheit - 32);  
}

Vòng lặp:

for (i = 0; i < 5; i++) {  
  x += i;  
}

Điều kiện:

if (time < 20) {  
  greeting = "Good day";  
} else {  
  greeting = "Good evening";  
}

* Quy tắc đối tượng

Quy tắc chung cho định nghĩa đối tượng:

* Đặt dấu ngoặc mở trên cùng dòng với tên đối tượng.
* Sử dụng dấu hai chấm cộng với một khoảng trắng giữa mỗi thuộc tính và giá trị của nó.
* Sử dụng dấu ngoặc kép quanh các giá trị chuỗi, không phải xung quanh các giá trị số.
* Không thêm dấu phẩy sau cặp giá trị thuộc tính cuối cùng.
* Đặt khung đóng trên một dòng mới, không có khoảng trắng ở đầu.
* Luôn kết thúc một định nghĩa đối tượng bằng dấu chấm phẩy.

Thí dụ

var person = {  
  firstName: "John",  
  lastName: "Doe",  
  age: 50,  
  eyeColor: "blue"  
};

Các đối tượng ngắn có thể được nén, trên một dòng, chỉ sử dụng khoảng trắng giữa các thuộc tính, như thế này:

var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};

* Độ dài dòng <80

Để dễ đọc, tránh các dòng dài hơn 80 ký tự.

* Quy ước đặt tên

Luôn sử dụng cùng một quy ước đặt tên cho tất cả mã của bạn. Ví dụ:

Tên biến và hàm được viết là camelCase

Các biến toàn cục được viết bằng UPPERCASE (Chúng tôi không, nhưng nó khá phổ biến)

Các hằng số (như PI) được viết bằng UPPERCASE

Bạn nên sử dụng gà mái , camelCase hoặc under\_scores trong tên biến?

Đây là một câu hỏi lập trình viên thường thảo luận. Câu trả lời phụ thuộc vào người bạn hỏi:

Các dấu gạch ngang trong HTML và CSS:

Các thuộc tính HTML5 có thể bắt đầu bằng dữ liệu- (số lượng dữ liệu, giá dữ liệu).

CSS sử dụng dấu gạch nối trong tên thuộc tính (cỡ chữ).

Dấu gạch dưới:

Nhiều lập trình viên thích sử dụng dấu gạch dưới (date\_of\_birth), đặc biệt là trong cơ sở dữ liệu SQL.

Underscores thường được sử dụng trong tài liệu PHP.

PascalCase:

PascalCase thường được các lập trình viên C ưa thích.

lạc đà:

camelCase được sử dụng bởi chính JavaScript, bởi jQuery và các thư viện JavaScript khác

* Đang tải JavaScript bằng HTML

Sử dụng cú pháp đơn giản để tải tập lệnh bên ngoài (thuộc tính type là không cần thiết):

<script src="myscript.js"></script>

* Truy cập các phần tử HTML

Hậu quả của việc sử dụng các kiểu HTML "không gọn gàng" có thể dẫn đến lỗi JavaScript.

Hai câu lệnh JavaScript này sẽ tạo ra các kết quả khác nhau:

var obj = getElementById("Demo")  
var obj = getElementById("demo")

* Phần mở rộng tập tin

Các tệp HTML phải có phần mở rộng .html (không phải .htm ).

Các tệp CSS nên có phần mở rộng .css .

Các tệp JavaScript nên có phần mở rộng .js .

* Sử dụng tên tệp chữ thường

Hầu hết các máy chủ web (Apache, Unix) đều phân biệt chữ hoa chữ thường về tên tệp:

london.jpg không thể được truy cập như London.jpg.

Các máy chủ web khác (Microsoft, IIS) không phân biệt chữ hoa chữ thường:

london.jpg có thể được truy cập như London.jpg hoặc london.jpg.

Nếu bạn sử dụng kết hợp chữ hoa và chữ thường, bạn phải cực kỳ nhất quán.

Nếu bạn chuyển từ một trường hợp không nhạy cảm, sang một máy chủ phân biệt chữ hoa chữ thường, ngay cả những lỗi nhỏ cũng có thể phá vỡ trang web của bạn.

Để tránh những vấn đề này, luôn luôn sử dụng tên tệp chữ thường (nếu có thể).

* Hiệu suất

Các quy ước mã hóa không được sử dụng bởi máy tính. Hầu hết các quy tắc có ít tác động đến việc thực hiện các chương trình.

Sự thụt lề và khoảng trắng thừa không đáng kể trong các tập lệnh nhỏ.

Đối với mã trong phát triển, khả năng đọc nên được ưu tiên. Kịch bản sản xuất lớn hơn nên được giảm thiểu.

#### JS Best Practices

Tránh các biến toàn cục, tránh new, tránh ==, tránheval()

* Tránh các biến toàn cầu

Giảm thiểu việc sử dụng các biến toàn cầu.

Điều này bao gồm tất cả các loại dữ liệu, đối tượng và chức năng.

Các biến và hàm toàn cầu có thể được ghi đè bằng các tập lệnh khác.

Thay vào đó, sử dụng các biến cục bộ và tìm hiểu cách sử dụng các bao đóng .

* Luôn khai báo các biến cục bộ

Tất cả các biến được sử dụng trong một hàm nên được khai báo là địa phương biến.

Các biến cục bộ phải được khai báo bằng var từ khóa hoặc lettừ khóa, nếu không chúng sẽ trở thành biến toàn cục.

* Tuyên bố trên

Đó là một thực hành mã hóa tốt để đặt tất cả các khai báo ở đầu mỗi tập lệnh hoặc hàm.

Điều này sẽ:

* Cung cấp mã sạch hơn
* Cung cấp một nơi duy nhất để tìm kiếm các biến cục bộ
* Giúp dễ dàng hơn để tránh các biến toàn cục (ngụ ý) không mong muốn
* Giảm khả năng khai báo lại không mong muốn

Điều này cũng áp dụng cho các biến vòng lặp:

* Khởi tạo biến

Đó là một thực hành mã hóa tốt để khởi tạo các biến khi bạn khai báo chúng.

Điều này sẽ:

* Cung cấp mã sạch hơn
* Cung cấp một nơi duy nhất để khởi tạo các biến
* Tránh các giá trị không xác định
* Không bao giờ khai báo các đối tượng số, chuỗi hoặc Boolean

Luôn coi số, chuỗi hoặc booleans là giá trị nguyên thủy. Không phải là đối tượng.

Khai báo các loại này dưới dạng đối tượng, làm chậm tốc độ thực thi và tạo ra các tác dụng phụ khó chịu

* Không sử dụng đối tượng mới ()

Sử dụng {}thay vìnew Object()

Sử dụng ""thay vìnew String()

Sử dụng 0thay vìnew Number()

Sử dụng falsethay vìnew Boolean()

Sử dụng []thay vìnew Array()

Sử dụng /()/thay vìnew RegExp()

Sử dụng function (){}thay vìnew Function()

* Cảnh giác với chuyển đổi loại tự độn

Coi chừng các số có thể vô tình được chuyển đổi thành chuỗi hoặc NaN(Không phải là Số).

JavaScript được gõ lỏng lẻo. Một biến có thể chứa các loại dữ liệu khác nhau và một biến có thể thay đổi loại dữ liệu của nó

* Sử dụng === So sánh

Các ==toán tử so sánh luôn luôn chuyển đổi (với các loại phù hợp) trước khi so sánh.

Các ===lực lượng toán tử so sánh các giá trị và loại

* Sử dụng mặc định tham số

Nếu một hàm được gọi với một đối số bị thiếu, giá trị của đối số bị thiếu được đặt thành undefined.

Các giá trị không xác định có thể phá vỡ mã của bạn. Đó là một thói quen tốt để gán giá trị mặc định cho các đối số

* Kết thúc chuyển mạch của bạn với mặc định

Luôn kết thúc switch câu nói của bạn bằng a default. Ngay cả khi bạn nghĩ rằng không cần nó.

* Tránh sử dụng eval ()

Các eval()chức năng được sử dụng để chạy văn bản dưới dạng mã. Trong hầu hết các trường hợp, không cần thiết phải sử dụng nó.

Bởi vì nó cho phép mã tùy ý được chạy, nó cũng đại diện cho một vấn đề bảo mật.

#### JS Mistakes

* Vô tình sử dụng toán tử chuyển nhượng

Các chương trình JavaScript có thể tạo ra kết quả không mong muốn nếu lập trình viên vô tình sử dụng toán tử gán ( =), thay vì toán tử so sánh ( ==) trong câu lệnh if.

Câu iflệnh này trả về false(như mong đợi) vì x không bằng 10

var x = 0;  
if (x == 10)

if Tuyên bố này trả về true(có thể không như mong đợi), vì 10 là đúng:

var x = 0;  
if (x = 10)

Câu if lệnh này trả về false(có thể không như mong đợi), vì 0 là sai:

var x = 0;  
if (x = 0)

* Mong đợi so sánh lỏng lẻo

Trong so sánh thường xuyên, loại dữ liệu không quan trọng. ifTuyên bố này trả về đúng:

var x = 10;  
var y = "10";  
if (x == y)

Trong so sánh nghiêm ngặt, loại dữ liệu có vấn đề. ifTuyên bố này trả về false:

Đó là một lỗi phổ biến để quên rằng các switchtuyên bố sử dụng so sánh nghiêm ngặt:

Điều này case switchsẽ hiển thị một cảnh báo

Điều này case switchsẽ không hiển thị một cảnh báo:

* Bổ sung và kết hợp khó hiểu

Ngoài ra là về việc thêm số .

Ghép là về việc thêm chuỗi .

Trong JavaScript cả hai hoạt động sử dụng cùng một +toán tử.

Do đó, việc thêm một số dưới dạng số sẽ tạo ra kết quả khác với việc thêm một số dưới dạng chuỗi

Phao hiểu lầm

Tất cả các số trong JavaScript được lưu trữ dưới dạng số dấu phẩy động 64 bit (Floats).

Tất cả các ngôn ngữ lập trình, bao gồm JavaScript, đều gặp khó khăn với các giá trị dấu phẩy động chính xác

* Phá vỡ một chuỗi JavaScript

JavaScript sẽ cho phép bạn chia một câu lệnh thành hai dòng

Nhưng, phá vỡ một câu lệnh ở giữa một chuỗi sẽ không hoạt động

Bạn phải sử dụng "dấu gạch chéo ngược" nếu bạn phải ngắt một câu lệnh trong chuỗi

* Đặt dấu chấm phẩy

Do dấu chấm phẩy bị đặt sai vị trí, khối mã này sẽ thực thi bất kể giá trị của x if (x == 19);

if (x == 19);  
{  
  // code block   
}

* Phá vỡ một tuyên bố trở lại

Đó là một hành vi JavaScript mặc định để tự động đóng một câu lệnh ở cuối dòng

JavaScript cũng sẽ cho phép bạn chia một câu lệnh thành hai dòng

* Giải trình

Nếu một tuyên bố không đầy đủ như:

Var

JavaScript sẽ cố gắng hoàn thành tuyên bố bằng cách đọc dòng tiếp theo:

power = 10;

Nhưng vì tuyên bố này đã hoàn tất:

Return

JavaScript sẽ tự động đóng nó như thế này:

return;

Điều này xảy ra vì các câu lệnh đóng (kết thúc) bằng dấu chấm phẩy là tùy chọn trong JavaScript.

JavaScript sẽ đóng câu lệnh return ở cuối dòng, bởi vì nó là một câu lệnh hoàn chỉnh.

Không bao giờ phá vỡ một tuyên bố trở lại.

* Truy cập mảng với chỉ mục được đặt tên

Nhiều ngôn ngữ lập trình hỗ trợ mảng với các chỉ mục được đặt tên.

Mảng với các chỉ mục được đặt tên được gọi là mảng kết hợp (hoặc băm).

JavaScript không hỗ trợ các mảng với các chỉ mục được đặt tên.

Trong JavaScript, mảng sử dụng các chỉ mục được đánh số :

Thí dụ

var person = [];  
person[0] = "John";  
person[1] = "Doe";  
person[2] = 46;  
var x = person.length;       // person.length will return 3  
var y = person[0];           // person[0] will return "John

Trong JavaScript, các đối tượng sử dụng các chỉ mục được đặt tên .

Nếu bạn sử dụng một chỉ mục được đặt tên, khi truy cập một mảng, JavaScript sẽ xác định lại mảng thành một đối tượng tiêu chuẩn.

Sau khi xác định lại tự động, các phương thức và thuộc tính mảng sẽ tạo ra kết quả không xác định hoặc không chính xác:

Thí dụ:

var person = [];  
person["firstName"] = "John";  
person["lastName"] = "Doe";  
person["age"] = 46;  
var x = person.length;      // person.length will return 0  
var y = person[0];          // person[0] will return undefined

* Kết thúc định nghĩa bằng dấu phẩy

Dấu phẩy dấu phẩy trong định nghĩa đối tượng và mảng là hợp pháp trong ECMAScript

Ví dụ đối tượng:

person = {firstName:"John", lastName:"Doe", age:46,}

Ví dụ mảng:

points = [40, 100, 1, 5, 25, 10,];

* Không xác định là không Null

Các đối tượng, biến, thuộc tính và phương thức JavaScript có thể undefined.

Ngoài ra, các đối tượng JavaScript trống có thể có giá trị null.

Điều này có thể làm cho một chút khó khăn để kiểm tra nếu một đối tượng trống

#### JS Performance

* Giảm hoạt động trong các vòng lặp

Vòng lặp thường được sử dụng trong lập trình.

Mỗi câu lệnh trong một vòng lặp, bao gồm câu lệnh for, được thực thi cho mỗi lần lặp của vòng lặp.

Các câu lệnh hoặc bài tập có thể được đặt bên ngoài vòng lặp sẽ làm cho vòng lặp chạy nhanh hơn.

Mã xấu truy cập thuộc tính độ dài của một mảng mỗi khi vòng lặp được lặp lại.

Mã tốt hơn truy cập thuộc tính độ dài bên ngoài vòng lặp và làm cho vòng lặp chạy nhanh hơn

* Giảm quyền truy cập DOM

Truy cập HTML DOM rất chậm, so với các câu lệnh JavaScript khác.

Nếu bạn muốn truy cập một phần tử DOM nhiều lần, hãy truy cập nó một lần và sử dụng nó như một biến cục bộ

* Giảm kích thước DOM

Giữ số lượng phần tử trong DOM DOM nhỏ.

Điều này sẽ luôn cải thiện tải trang và tăng tốc độ kết xuất (hiển thị trang), đặc biệt là trên các thiết bị nhỏ hơn.

Mọi nỗ lực tìm kiếm DOM (như getElementsByTagName) sẽ được hưởng lợi từ một DOM nhỏ hơn.

* Tránh các biến không cần thiết

Đừng tạo các biến mới nếu bạn không có kế hoạch lưu các giá trị.

Thường thì bạn có thể thay thế mã như thế này

var fullName = firstName + " " + lastName;

document.getElementById("demo").innerHTML = fullName;

Với cái này:

document.getElementById("demo").innerHTML = firstName + " " + lastName

* Trì hoãn tải JavaScript

Đặt tập lệnh của bạn ở dưới cùng của thân trang cho phép trình duyệt tải trang trước.

Trong khi tập lệnh đang tải xuống, trình duyệt sẽ không bắt đầu bất kỳ tải xuống nào khác. Ngoài ra, tất cả các hoạt động phân tích cú pháp và kết xuất có thể bị chặn.

Một cách khác là sử dụng defer="true"trong thẻ script. Thuộc tính defer xác định rằng tập lệnh sẽ được thực thi sau khi trang đã phân tích xong, nhưng nó chỉ hoạt động cho các tập lệnh bên ngoài

* Tránh sử dụng với

Tránh sử dụng withtừ khóa. Nó có ảnh hưởng tiêu cực đến tốc độ. Nó cũng làm tăng phạm vi JavaScript.

Các withtừ khóa không được phép trong chế độ nghiêm ngặt

#### JS Reserved Words

#### JS Versions ES5

* ECMAScript 5 là gì?

ECMAScript 5 còn được gọi là ES5 và ECMAScript 2009

Chương này giới thiệu một số tính năng quan trọng nhất của ES5.

* Tính năng ECMAScript 5

Đây là những tính năng mới được phát hành năm 2009:

các "use strict"Chỉ thị

String.trim()

Array.isArray()

Array.forEach()

Array.map()

Array.filter()

Array.reduce()

Array.reduceRight()

Array.every()

Array.some()

Array.indexOf()

Array.lastIndexOf()

JSON.parse()

JSON.stringify()

Date.now()

Tài sản Getters và Setters

Phương thức thuộc tính đối tượng mới

* ECMAScript 5 Thay đổi cú pháp

Truy cập thuộc tính [] trên chuỗi

Trailing dấu phẩy trong mảng và đối tượng bằng chữ

Chuỗi ký tự nhiều dòng

Các từ dành riêng như tên tài sản

* Chỉ thị "sử dụng nghiêm ngặt"

"use strict" định nghĩa rằng mã JavaScript phải được thực thi trong "chế độ nghiêm ngặt".

Với chế độ nghiêm ngặt, bạn có thể, ví dụ, không sử dụng các biến không được khai báo

* Chuỗi.trim ()

String.trim() loại bỏ khoảng trắng từ cả hai phía của một chuỗi.

* Array.isArray ()

Các isArray()phương pháp kiểm tra xem một đối tượng là một mảng.

* Mảng.forEach ()

Các forEach()phương pháp gọi một chức năng một lần cho mỗi phần tử mảng.

* Mảng.map ()
* Array.filter ()
* Array.reduce ()
* Array.reduceRight ()
* Array.every ()
* Mảng.some ()
* Mảng.indexOf ()

Tìm kiếm một mảng cho một giá trị phần tử và trả về vị trí của nó.

* Array.lastIndexOf ()

Array.lastIndexOf()là giống như Array.indexOf(), nhưng tìm kiếm từ cuối mảng.

* JSON.parse ()

Một cách sử dụng phổ biến của JSON là nhận dữ liệu từ máy chủ web.

Hãy tưởng tượng bạn đã nhận được chuỗi văn bản này từ một máy chủ web

Hàm JavaScript JSON.parse()được sử dụng để chuyển đổi văn bản thành đối tượng JavaScript

* JSON.opesify ()

Một cách sử dụng phổ biến của JSON là gửi dữ liệu đến máy chủ web.

Khi gửi dữ liệu đến máy chủ web, dữ liệu phải là một chuỗi.

Hãy tưởng tượng chúng ta có đối tượng này trong JavaScript:

var obj = {"name":"John", "age":30, "city":"New York"};

Sử dụng hàm JavaScript JSON.stringify()để chuyển đổi nó thành một chuỗi.

var myJSON = JSON.stringify(obj);

Kết quả sẽ là một chuỗi theo ký hiệu JSON.

myJSON hiện là một chuỗi và sẵn sàng được gửi đến máy chủ

* Ngày.now ()

Date.now() trả về số mili giây kể từ ngày số không (ngày 1 tháng 1 năm 1970 00:00:00 UTC).

Date.now()trả về giống như getTime () được thực hiện trên một Dateđối tượng.

Tìm hiểu thêm trong Ngày JS

* Tài sản Getters và Setters

ES5 cho phép bạn xác định các phương thức đối tượng với một cú pháp trông giống như nhận hoặc đặt thuộc tính.

* Phương thức thuộc tính đối tượng mới

Object.defineProperty() là một phương thức Object mới trong ES5.

Nó cho phép bạn xác định một thuộc tính đối tượng và / hoặc thay đổi giá trị và / hoặc siêu dữ liệu của thuộc tính.

* Quyền truy cập thuộc tính trên chuỗi

Các charAt()phương thức trả về các nhân vật tại một chỉ số nào đó (vị trí) trong một chuỗi:

ECMAScript 5 cho phép truy cập thuộc tính trên chuỗi

* Dấu phẩy

ECMAScript 5 cho phép dấu phẩy trong các định nghĩa đối tượng và mảng

* Đối tượng JSON:

// Allowed:

var person = '{"firstName":"John", "lastName":"Doe", "age":46}'

JSON.parse(person)

// Not allowed:

var person = '{"firstName":"John", "lastName":"Doe", "age":46,}'

JSON.parse(person)

* Mảng JSON:

// Allowed:

points = [40, 100, 1, 5, 25, 10]

// Not allowed:

points = [40, 100, 1, 5, 25, 10,]

* Chuỗi trên nhiều dòng

ECMAScript 5 cho phép chuỗi ký tự trên nhiều dòng nếu thoát bằng dấu gạch chéo ngược

* Từ dành riêng như tên tài sản

ECMAScript 5 cho phép các từ dành riêng làm tên thuộc tính

* Hỗ trợ trình duyệt cho ES5 (ECMAScript 5)

Chrome 23, IE 10 và Safari 6 là những trình duyệt đầu tiên hỗ trợ đầy đủ ECMAScript 5

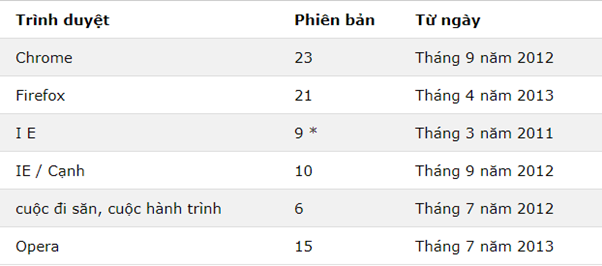
#### Js Version

* Hỗ trợ trình duyệt

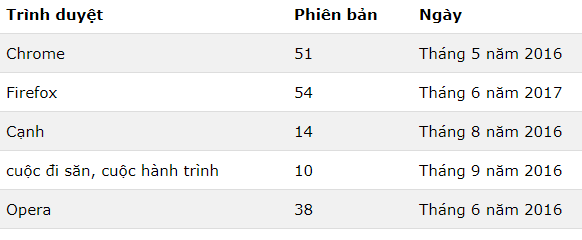
ECMAScript 3 được hỗ trợ đầy đủ trong tất cả các trình duyệt.

ECMAScript 5 được hỗ trợ đầy đủ trong tất cả các trình duyệt hiện đại .

* Hỗ trợ trình duyệt cho ES5 (2009)



* Hỗ trợ trình duyệt cho ES6 (ECMAScript 2015)



* Hỗ trợ trình duyệt cho ES7 (ECMAScript 2016)

Trình duyệt Phiên bản Ngày

Chrome 68 Tháng 5 năm 2018

Opera 55 Tháng 8 năm 2018

* JavaScript / ECMAScript

JavaScript được phát triển cho Netscape. Netscape 2 là trình duyệt đầu tiên chạy JavaScript.

Sau Netscape, nền tảng Mozilla tiếp tục phát triển JavaScript cho trình duyệt Firefox.

Phiên bản JavaScript mới nhất là 1.8.5. (Giống hệt với ECMAScript 5).

ECMAScript được phát triển bởi ECMA International sau khi tổ chức áp dụng JavaScript.

Phiên bản đầu tiên của ECMAScript được phát hành vào năm 1997

#### JS Version ES6

* ECMAScript 6 là gì?

ECMAScript 6 còn được gọi là ES6 và ECMAScript 2015.

Một số người gọi nó là JavaScript 6.

Chương này sẽ giới thiệu một số tính năng mới trong ES6.

* JavaScript let
* JavaScript const
* Hàm mũi tên JavaScript
* Các lớp học JavaScript
* Giá trị tham số mặc định
* Array.find()
* Array.findIndex()
* Số mũ ( \*\*) (EcmaScript 2016)
* Hỗ trợ trình duyệt cho ES6 (ECMAScript 2015)

Safari 10 và Edge 14 là những trình duyệt đầu tiên hỗ trợ đầy đủ ES6

* JavaScript để

Câu letlệnh cho phép bạn khai báo một biến có phạm vi khối.

JavaScript const

Câu constlệnh cho phép bạn khai báo một hằng số (một biến JavaScript có giá trị không đổi).

Các hằng số tương tự để cho các biến, ngoại trừ giá trị không thể thay đổi.

Chức năng mũi tên

Các hàm mũi tên cho phép một cú pháp ngắn để viết biểu thức hàm.

Các chức năng mũi tên không có riêng của họ this. Chúng không phù hợp để xác định các phương thức đối tượng .

Các chức năng mũi tên không được nâng lên. Chúng phải được xác định trước khi chúng được sử dụng.

Sử dụng const là an toàn hơn so với sử dụng var, bởi vì một biểu thức hàm luôn có giá trị không đổi.

Bạn chỉ có thể bỏ qua returntừ khóa và dấu ngoặc nhọn nếu hàm là một câu lệnh đơn. Vì điều này, nó có thể là một thói quen tốt để luôn giữ chúng

Bạn không cần functiontừ khóa, returntừ khóa và dấu ngoặc nhọn .

Các lớp học

ES6 giới thiệu các lớp học.Một lớp là một loại hàm, nhưng thay vì sử dụng từ khóa functionđể khởi tạo nó, chúng ta sử dụng từ khóa classvà các thuộc tính được gán bên trong một constructor()phương thức.

Sử dụng từ khóa classđể tạo một lớp và luôn thêm phương thức constructor.

Phương thức constructor được gọi mỗi khi đối tượng lớp được khởi tạo.

Thí dụ

Một định nghĩa lớp đơn giản cho một lớp có tên car:

class Car {

constructor(brand) {

this.carname = brand;

}

}

Giá trị tham số mặc định

ES6 cho phép các tham số chức năng có các giá trị mặc định.

Array.find ()

Các find()phương thức trả về giá trị của các phần tử mảng đầu tiên mà đi một chức năng kiểm tra.

Array.find Index ()

Các findIndex()phương thức trả về chỉ số của phần tử mảng đầu tiên mà đi một chức năng kiểm tra.

Thuộc tính số mới

ES6 đã thêm các thuộc tính sau vào đối tượng Number:

EPSILON

MIN\_SAFE\_INTEGER

MAX\_SAFE\_INTEGER

Phương pháp số mới

ES6 đã thêm 2 phương thức mới vào đối tượng Number:

Number.isInteger()

Number.isSafeInteger()

Phương thức Number.isInteger ()

Các Number.isInteger() phương thức trả về truenếu đối số là một số nguyên.

Phương thức Number.isSafeInteger ()

Số nguyên an toàn là số nguyên có thể được biểu diễn chính xác dưới dạng số chính xác kép.

Các Number.isSafeInteger()phương thức trả về truenếu đối số là một số nguyên an toàn

Phương pháp toàn cầu mới

ES6 cũng đã thêm 2 phương thức số toàn cầu mới:

isFinite()

isNaN()

Phương thức isFinite ()

isFinite()Phương thức toàn cục trả về falsenếu đối số là Infinityhoặc NaN

#### JS JSON

JSON là một định dạng để lưu trữ và vận chuyển dữ liệu.

JSON thường được sử dụng khi dữ liệu được gửi từ máy chủ đến trang web.

* JSON là gì?

JSON là viết tắt của J ava S cript O bject N otation

JSON là một định dạng trao đổi dữ liệu nhẹ

JSON là ngôn ngữ độc lập \*

JSON là "tự mô tả" và dễ hiểu

\* Cú pháp JSON có nguồn gốc từ cú pháp ký hiệu đối tượng JavaScript, nhưng định dạng JSON chỉ là văn bản. Mã để đọc và tạo dữ liệu JSON có thể được viết bằng bất kỳ ngôn ngữ lập trình nào.

* Ví dụ JSON

Cú pháp JSON này xác định một đối tượng nhân viên: một mảng gồm 3 bản ghi nhân viên (đối tượng)

Định dạng JSON Đánh giá các đối tượng JavaScript

Định dạng JSON hoàn toàn giống với mã để tạo các đối tượng JavaScript.

Do sự giống nhau này, một chương trình JavaScript có thể dễ dàng chuyển đổi dữ liệu JSON thành các đối tượng JavaScript gốc.

* Quy tắc cú pháp JSON

Dữ liệu nằm trong cặp tên / giá trị

Dữ liệu được phân tách bằng dấu phẩy

Niềng răng giữ đồ vật

Dấu ngoặc vuông giữ mảng

* Dữ liệu JSON - Tên và giá trị

Dữ liệu JSON được viết dưới dạng cặp tên / giá trị, giống như các thuộc tính đối tượng JavaScript.

Một cặp tên / giá trị bao gồm một tên trường (trong dấu ngoặc kép), theo sau là dấu hai chấm, theo sau là một giá trị:

"firstName":"John"

* Đối tượng JSON

Các đối tượng JSON được viết bên trong dấu ngoặc nhọn.

Giống như trong JavaScript, các đối tượng có thể chứa nhiều cặp tên / giá trị:

{"firstName":"John", "lastName":"Doe"}

* Mảng JSON

Mảng JSON được viết bên trong dấu ngoặc vuông.

Giống như trong JavaScript, một mảng có thể chứa các đối tượng:

"employees":[

{"firstName":"John", "lastName":"Doe"},

{"firstName":"Anna", "lastName":"Smith"},

{"firstName":"Peter", "lastName":"Jones"}

]

Trong ví dụ trên, đối tượng "nhân viên" là một mảng. Nó chứa ba đối tượng.

Mỗi đối tượng là một bản ghi của một người (có tên và họ)

* Chuyển đổi một văn bản JSON thành một đối tượng JavaScript

Một cách sử dụng phổ biến của JSON là đọc dữ liệu từ máy chủ web và hiển thị dữ liệu trong trang web.

Để đơn giản, điều này có thể được thể hiện bằng cách sử dụng một chuỗi làm đầu vào.

Đầu tiên, tạo một chuỗi JavaScript chứa cú pháp JSON:

var text = '{ "employees" : [' +

'{ "firstName":"John" , "lastName":"Doe" },' +

'{ "firstName":"Anna" , "lastName":"Smith" },' +

'{ "firstName":"Peter" , "lastName":"Jones" } ]}';

Sau đó, sử dụng hàm dựng sẵn JavaScript JSON.parse()để chuyển đổi chuỗi thành đối tượng JavaScript:

var obj = JSON.parse(text);

Cuối cùng, sử dụng đối tượng JavaScript mới trong trang của bạn

### Biểu mẫu và xác thực dữ liệu nhập

#### JS Forms

HTML forms được sử dụng để người dùng nhập liệu, và thông thường được kiểm tra lại bằng JavaScript. Ví dụ nếu trường văn bẳn có fname là rỗng, JavaScript có thể kiểm tra và đưa ra thông điệp cảnh báo. Ví dụ một hàm kiểm tra dữ liệu có thể được viết như sau:

function validateForm() {  
  var x = document.forms["myForm"]["fname"].value;  
  if (x == "") {  
    alert("Name must be filled out");  
    return false;  
  }  
}

Hàm trên có thể được gọi khi người dùng bấm xác việc hoàn tất nhập liệu,

<form name="myForm" action="/action\_page.php" **onsubmit="return validateForm()"** method="post">  
Name: <input type="text" name="fname">  
<input type="submit" value="Submit">  
</form>

#### Forms API

JavaScript cung cấp 2 hàm là checkValidity() và setCustomValidity() để kiểm tra sự hợp lệ của dữ liệu. Hai hàm này sẽ kết hợp với các thuộc tính ràng buộc dữ liệu liệt kê trong bảng sau:

|  |  |
| --- | --- |
| Thuộc tính | Mô tả |
| disabled | Đối tượng nhập liệu không thể sử dụng |
| max | Giá trị lớn nhất cho phép |
| min | Giá trị nhỏ nhất cho phép |
| pattern | Khuôn mẫu của dữ liệu |
| required | Bắt buộc nhập dữ liệu |
| type | Kiểu dữ liệu |

Khi các thành phần nhập liệu được quy đinh các thuộc tính trên, trong hàm kiểm tra dữ liệu với JavaScript chỉ cần gọi hàm checkValidity(). Nếu dữ liệu hợp lệ sẽ trả về đúng (true), hoặc khong hợp lệ sẽ trả về sai (false) từ đó là cơ sở để đưa ra các cảnh báo hoặc nhắc nhở cần thiết. Ví dụ:

<input id="id1" type="number" min="100" max="300" required>  
<button onclick="myFunction()">OK</button>  
<p id="demo"></p>  
<script>  
function myFunction() {  
  var inpObj = document.getElementById("id1");  
  if (!inpObj.checkValidity()) {  
    document.getElementById("demo").innerHTML = inpObj.validationMessage;  
  }  
}  
</script>

Đối với các kiểu dữ liệu phức tạp hơn, có thể kết hợp sử dụng thêm các thuộc tính của thuộc tính validity. Các thuộc tính của thuộc tính validity được liệt kê trong bảng dưới đây:

|  |  |
| --- | --- |
| Thuộc tính | Ý nghĩa |
| customError | Bằng đúng (true) nếu được thiết lập cảnh báo tủy chỉnh |
| patternMismatch | Bằng đúng (true) nếu dữ liệu nhập sai với khuôn mẫu |
| rangeOverflow | Bằng đúng (true) nếu dữ liệu lớn hơn giá trị cho phép |
| rangeUnderflow | Bằng đúng (true) nếu dữ liệu nhỏ hơn giá trị cho phép |
| stepMismatch | Bằng đúng (true) nếu dữ liệu sai thuộc tính bước (step) |
| tooLong | Bằng đúng (true) nếu giá trị vượt quá chiều dài cho phép |
| typeMismatch | Bằng đúng (true) dữ liệu bị sai kiểu |
| valueMissing | Bằng đúng (true) nếu chưa có dữ liệu |
| valid | Bằng đúng (true) nếu dữ liệu là hợp lệ |

Ví dụ sau minh họa cách sử dụng thuộc tính trên để kiểm tra dữ liệu người dùng nhập có nằm trong khoảng cho phép (từ 20 đến 100) hay không.

<input id="id1" type="number" max="100" min="20">  
<button onclick="myFunction()">OK</button>  
<p id="demo"></p>  
<script>  
function myFunction() {  
  var txt = "";  
  if (document.getElementById("id1").validity.rangeOverflow) {  
    txt = "Value too large";  
  }

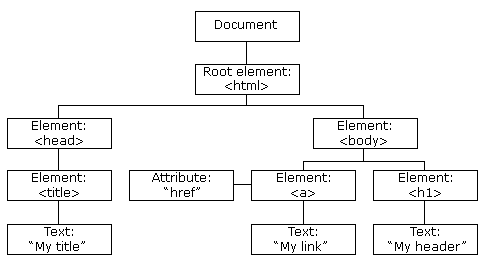
else if (document.getElementById("id1").validity.rangeUnderflow) {  
    txt = "Value too small";  
  }  
  document.getElementById("demo").innerHTML = txt;  
}  
</script>

### Mô hình đối tượng tài liệu (DOM – Document Oject Model)

#### DOM Intro

Khi một trang web được tải, trình duyệt tạo ra một D ocument O bject M odel trang.

Các HTML DOM mô hình được xây dựng như một cây của các đối tượng :



Với mô hình đối tượng, JavaScript có tất cả sức mạnh cần thiết để tạo HTML động:

JavaScript có thể thay đổi tất cả các yếu tố HTML trong trang

JavaScript có thể thay đổi tất cả các thuộc tính HTML trong trang

JavaScript có thể thay đổi tất cả các kiểu CSS trong trang

JavaScript có thể xóa các thành phần và thuộc tính HTML hiện có

JavaScript có thể thêm các yếu tố và thuộc tính HTML mới

JavaScript có thể phản ứng với tất cả các sự kiện HTML hiện có trong trang

JavaScript có thể tạo các sự kiện HTML mới trong trang

DOM là gì?

DOM là một tiêu chuẩn W3C (World Wide Web Consortium).

DOM định nghĩa một tiêu chuẩn để truy cập tài liệu:

"Mô hình đối tượng tài liệu W3C (DOM) là giao diện trung lập về ngôn ngữ và nền tảng cho phép các chương trình và tập lệnh truy cập động và cập nhật nội dung, cấu trúc và kiểu tài liệu."

Tiêu chuẩn W3C DOM được tách thành 3 phần khác nhau:

Core DOM - mô hình chuẩn cho tất cả các loại tài liệu

XML DOM - mô hình chuẩn cho các tài liệu XML

HTML DOM - mô hình chuẩn cho các tài liệu HTML

HTML DOM là gì?

HTML DOM là một mô hình đối tượng tiêu chuẩn và giao diện lập trình cho HTML. Nó định nghĩa:

Các phần tử HTML làm đối tượng

Các thuộc tính của tất cả các thành phần HTML

Các phương thức để truy cập tất cả các phần tử HTML

Các sự kiện cho tất cả các yếu tố HTML

Nói cách khác: HTML DOM là một tiêu chuẩn cho cách nhận, thay đổi, thêm hoặc xóa các thành phần HTML.

#### DOM Methods

Các phương thức HTML DOM là các hành động bạn có thể thực hiện (trên Các phần tử HTML).

Thuộc tính HTML DOM là các giá trị (của các thành phần HTML) mà bạn có thể đặt hoặc thay đổi.

Giao diện lập trình DOM

* HTML DOM có thể được truy cập bằng JavaScript (và với các ngôn ngữ lập trình khác).
* Trong DOM, tất cả các phần tử HTML được định nghĩa là các đối tượng .
* Giao diện lập trình là các thuộc tính và phương thức của từng đối tượng.

Ví dụ sau thay đổi nội dung (phần innerHTML) của <p>phần tử bằng id="demo":

<html>

<body>

<p id="demo"></p>

<script>

document.getElementById("demo").innerHTML = "Hello World!";

</script>

</body>

</html>

trong ví dụ trên, getElementByIdlà một phương thức , trong khi innerHTMLlà một thuộc tính

Phương thức getEuityById

Cách phổ biến nhất để truy cập phần tử HTML là sử dụng idphần tử.

Trong ví dụ trên, getElementByIdphương thức được sử dụng id="demo"để tìm phần tử.

#### DOM Document

DOM document: có nhiệm vụ lưu trữ toàn bộ các thành phần trong tài liệu của website

#### DOM Elements

Tìm thẻ HTML theo ID

Để truy xuất tới một thẻ HTML theo ID ta sử dụng cú pháp sau:

var element = document.getElementById('idname');

vi du :

// Lấy thẻ input

var element = document.getElementById('website');

// Lấy giá trị của thẻ input

document.write(element.value);

Tìm thẻ HTML theo tên của thẻ HTML

Tên thẻ HTML chính là tên các thẻ như p, a, div, ... Và ta sẽ truy xuất tới nó bằng cú pháp sau:

varelement = document.getElementsByTagName('tagname');

vi du :

// Lấy thẻ input

var element = document.getElementsByTagName('input');

// Lấy giá trị của thẻ input

document.write(element[0].value);

Trong ví dụ này có sự khác biệt đó là đoạn code element[0].value. Tại sao lại như vậy? Như bạn biết trong một trang web có thể có nhiều thẻ HTML giống nhau (ví dụ có hai thẻ div) nên hàm getElementsByTagName() sẽ trả về một mảng các object (xem bài mảng trong javascript) chứ không phải là một object nữa, chính vì vậy ta sẽ lấy input thứ nhất nên truyền số 0 vào.

Tìm thẻ HTML theo tên class

Để tìm các thẻ HTML có class nào đó thì ta dùng cú pháp sau:

var element = document.getElementsByClassName('input');

Tương tự như tìm theo tên thẻ HTML thì tìm theo tên class sẽ trả về một mảng các object nên bạn sẽ phải sử dụng cú pháp truy xuất mảng để chọn đúng đối tượng muốn lấy

vi du :

// Lấy thẻ input

var element = document.getElementsByClassName('website');

// Lấy giá trị của thẻ input

document.write(element[0].value);

Tìm thẻ HTML theo cú pháp của Selector CSS

Khi chọn các thẻ HTML theo class thường sẽ trả về hàng loạt các kết quả, như vậy đôi khi sẽ có những kết quả mà ta không mong đợi. Chính vì vậy javascript DOM có một phương thức kết hợp với CSS Selector để truy vấn có độ chính xác cao hơn.

Và đây là cú pháp sử dụng:

var element = document.querySelectorAll("selector.css");

#### DOM HTML

Thay đổi và lấy nội dung bên trong thẻ HTML

Để lấy nội dung bên trong một thẻ HTML thì chúng ta sử dụng cú pháp như sau:

var html = document.getElementById("content").innerHTML

Và để thay đổi nội dung cho một thẻ HTML thì ta dùng cú pháp sau:

var html = document.getElementById("content").innerHTML = "<h1>Nội dung</h1>";

Ví dụ: Trong ví dụ này ta sẽ viết chương trình thay đổi nội dung HTML của một thẻ DIV và lấy nội dung bên trong của một thẻ DIV

<html>

  <body>

    <script language="javascript">

      // Hàm thiết lập nội dung cho thẻ div#content

      function set\_content()

      {

        document.getElementById("content").innerHTML = "<h1>Nội dung đã được thay đổi</h1>";

      }

      // Hàm lấy nội dung cho thẻ div#content

      function get\_content()

      {

        var html = document.getElementById("content").innerHTML;

        alert("Nội dung cần lấy là: " + html);

      }

    </script>

    <div id="content">Nội dung của thẻ DIV</div>

    <input type="button" value="Lấy nội dung" id="get\_content" onclick="get\_content()" />

    <input type="button" value="Thay đổi nội dung" id="set\_content" onclick="set\_content()" />

  </body>

Thay đổi và lấy giá trị thuộc tính thẻ HTML bằng Javascript

Để thay đổi giá trị của một thuộc tính HTML bất kì thì ta sử dụng cú pháp sau:

document.getElementById("element").attributeName = "new value";

Để lấy giá trị của một thuộc tính HTML ta sử dụng cú pháp sau:

var value = document.getElementById("element").attributeName;

Từ đây có thể suy ra rằng trong Javascript để thiết lập (set) và lấy (get) thì sử dụng chung một cú pháp, chỉ khác nhau ở chỗ gán bằng và không có gán bằng.

Ví dụ: Xây dựng chương trình khi click vào một button thì chuyển nó thành textbox, và tiếp tục click vào textbox thì sẽ đổi thành button

<html>

    <body>

        <script language="javascript">

          function change()

          {

             // Lấy đối tượng

             var object = document.getElementById("object");

           // lấy thuộc tính type

             var type = object.type;

             // kiểm tra thuộc tính type và thay đổi

            if (type == "button"){

               object.type = 'text';

            }

            else{

                object.type = "button";

            }

          }

        </script>

        <input type="button" value="CLick me" onclick="change()" id="object" />

    </body>

</html>

#### DOM CSS

Thay đổi CSS bằng Javascript

Style bản chất nó cũng là một thuộc tính của các thẻ HTML nhưng bạn không thể sử dụng DOM HTML để thiết lập hay xóa bỏ CSS được mà phải thông qua một đối tượng biệt khác đó là style.

Đối tượng style này sẽ chứa tất cả các thuộc tính của CSS và chúng ta sẽ dễ dàng thao tác với chúng bằng cú pháp riêng, và như thường lệ chúng ta có hai thao tác chính đó là thiết lập CSS và lấy giá trị CSS hiện tại.

Cú pháp thiết lập CSS bằng Javascript:

document.getElementById("object").style.cssName = 'something';

Cú pháp lấy giátrị CSS bằng Javascript:

value = document.getElementById("object").style.cssName;var

Trường hợp thuộc tính có dấu gạch ngang như: font-size, line-height, margin-bottom thì thuộc tính đó trong style sẽ có tên là fontSize, lineHeight, marginBottom ,nghĩa là sẽ bỏ đi dấu gạch ngang và viết hoa ký tự đầu tiên của chữ thứ hai.

Lưu ý rằng có những thuộc tính nếu bạn chưa thiết lập CSS cho nó thì khi bạn lấy giá trị sẽ là một giá trị rỗng

#### DOM Animations

Tạo một thùng chứa hoạt hình

Tất cả các hình ảnh động nên liên quan đến một yếu tố container.

Thí dụ

**<div** id ="container"**>**

**<div** id ="animate"**>**My animation will go here**</div>**

**</div>**

Phần tử container nên được tạo với style = " position: relative".

Phần tử hoạt hình nên được tạo với style = " position: absolute".

vi du

<!Doctype html**>**

**<html>**

**<style>**

#container {

  width: 400px;

  height: 400px;

  position: relative;

  background: yellow;

}

#animate {

  width: 50px;

  height: 50px;

  position: absolute;

  background: red;

}

**</style>**

**<body>**

**<h2>**My First JavaScript Animation**</h2>**

**<div** id="container"**>**

**<div** id="animate"**></div>**

**</div>**

**</body>**

**</html>**

Mã hoạt hình

Hoạt ảnh JavaScript được thực hiện bằng cách lập trình thay đổi dần dần theo kiểu của một phần tử.

Những thay đổi được gọi bởi một bộ đếm thời gian. Khi khoảng thời gian hẹn giờ nhỏ, hình ảnh động trông liên tục.

Mã cơ bản là:

Thí dụ

**var** id = setInterval(frame, 5);

**function** frame() {

**if** (/\* test for finished \*/) {

    clearInterval(id);

  } **else** {

    /\* code to change the element style \*/

  }

}

Tạo hoạt hình bằng JavaScript

vi du

<!DOCTYPE html**>**

**<html>**

**<style>**

#container {

  width: 400px;

  height: 400px;

  position: relative;

  background: yellow;

}

#animate {

  width: 50px;

  height: 50px;

  position: absolute;

  background-color: red;

}

**</style>**

**<body>**

**<p><button** onclick="myMove()"**>**Click Me**</button></p>**

**<div** id ="container"**>**

**<div** id ="animate"**></div>**

**</div>**

**<script>**

function myMove() {

  var elem = document.getElementById("animate");

  var pos = 0;

  var id = setInterval(frame, 5);

  function frame() {

    if (pos == 350) {

      clearInterval(id);

    } else {

      pos++;

      elem.style.top = pos + "px";

      elem.style.left = pos + "px";

    }

  }

}

**</script>**

**</body>**

**</html>**

#### DOM Events

JavaScript có thể được thực thi khi xảy ra sự kiện, như khi người dùng nhấp vào phần tử HTML.

Để thực thi mã khi người dùng nhấp vào một phần tử, hãy thêm mã JavaScript vào thuộc tính sự kiện HTML:

onclick=JavaScript

Ví dụ về các sự kiện HTML:

Khi người dùng nhấp chuột

Khi một trang web đã được tải

Khi một hình ảnh đã được tải

Khi chuột di chuyển qua một phần tử

Khi một trường đầu vào được thay đổi

Khi một biểu mẫu HTML được gửi

Khi người dùng vuốt một phím

sự kiên onload và onunload:

Các sự kiện onloadvà onunloadđược kích hoạt khi người dùng vào hoặc rời trang.

Sự onloadkiện này có thể được sử dụng để kiểm tra loại trình duyệt và phiên bản trình duyệt của khách truy cập và tải phiên bản phù hợp của trang web dựa trên thông tin.

Các sự kiện onloadvà onunloadcó thể được sử dụng để đối phó với cookie.

Sự kiện trao đổi

Sự onchange kiện này thường được sử dụng kết hợp với xác nhận các trường đầu vào.

Dưới đây là một ví dụ về cách sử dụng onchange. Các upperCase() chức năng sẽ được gọi khi người dùng thay đổi nội dung của một lĩnh vực đầu vào.

Sự kiện onmouseover và onmouseout

Các sự kiện onmouseovervà onmouseoutcó thể được sử dụng để kích hoạt một chức năng khi người dùng kết hợp hoặc thoát khỏi một phần tử HTML

#### DOM Event Listener

Phương thức addEventListener ()

Thí dụ

Thêm một trình lắng nghe sự kiện kích hoạt khi người dùng nhấp vào nút:

document.getElementById("myBtn").addEventListener("click", displayDate);

Các addEventListener()phương pháp gắn một event handler để các yếu tố quy định.

Các addEventListener()phương pháp gắn một event handler cho một yếu tố mà không ghi đè lên xử lý sự kiện hiện có.

Bạn có thể thêm nhiều trình xử lý sự kiện vào một phần tử.

Bạn có thể thêm nhiều trình xử lý sự kiện cùng loại vào một thành phần, tức là hai sự kiện "nhấp chuột".

Bạn có thể thêm trình lắng nghe sự kiện vào bất kỳ đối tượng DOM nào, không chỉ các phần tử HTML. tức là đối tượng cửa sổ.

Các addEventListener()phương pháp làm cho nó dễ dàng hơn để kiểm soát cách sự kiện này phản ứng với bọt.

Khi sử dụng addEventListener()phương thức, JavaScript được tách ra khỏi đánh dấu HTML, để dễ đọc hơn và cho phép bạn thêm trình lắng nghe sự kiện ngay cả khi bạn không kiểm soát đánh dấu HTML.

Bạn có thể dễ dàng loại bỏ một người nghe sự kiện bằng cách sử dụng removeEventListener()phương pháp.

Cú pháp

element.addEventListener(event, function, useCapture);

Tham số đầu tiên là loại sự kiện (như " click" hoặc " mousedown" hoặc bất kỳ Sự kiện DOM DOM nào khác .)

Tham số thứ hai là hàm chúng ta muốn gọi khi sự kiện xảy ra.

Tham số thứ ba là một giá trị boolean xác định xem nên sử dụng bong bóng sự kiện hay chụp sự kiện. Tham số này là tùy chọn.

Lưu ý rằng bạn không sử dụng tiền tố "bật" cho sự kiện; sử dụng " click" thay vì " onclick".

Thêm một Trình xử lý sự kiện cho một phần tử

Thí dụ

Thông báo "Xin chào thế giới!" khi người dùng nhấp vào một yếu tố:

element.addEventListener("click", function(){ alert("Hello World!"); });

Thêm nhiều Trình xử lý sự kiện vào cùng một phần tử

Các addEventListener()phương pháp cho phép bạn thêm nhiều sự kiện để cùng một nguyên tố, mà không ghi đè lên các sự kiện hiện có

Thêm một Trình xử lý sự kiện vào cửa sổ Đối tượng

Các addEventListener()phương pháp cho phép bạn thêm người nghe sự kiện trên bất kỳ đối tượng DOM HTML như các phần tử HTML, tài liệu HTML, các đối tượng cửa sổ, hoặc các đối tượng khác có hỗ trợ sự kiện, giống như xmlHttpRequestđối tượng

Thông số vượt qua

Khi truyền các giá trị tham số, sử dụng "hàm ẩn danh" gọi hàm được chỉ định với các tham số:

Thí dụ

element.addEventListener("click", function(){ myFunction(p1, p2); });

Có hai cách truyền bá sự kiện trong HTML DOM, tạo bọt và bắt.

Tuyên truyền sự kiện là một cách xác định thứ tự phần tử khi một sự kiện xảy ra. Nếu bạn có phần tử <p> bên trong phần tử <div> và người dùng nhấp vào phần tử <p>, thì sự kiện "nhấp chuột" của phần tử nào sẽ được xử lý trước?

Trong bong bóng , sự kiện phần lớn bên trong được xử lý trước và sau đó là bên ngoài: sự kiện nhấp chuột của phần tử <p> được xử lý trước, sau đó là sự kiện nhấp chuột của phần tử <div>.

Khi nắm bắt sự kiện bên ngoài của hầu hết các phần tử được xử lý trước và sau đó bên trong: sự kiện nhấp chuột của phần tử <div> sẽ được xử lý trước, sau đó là sự kiện nhấp chuột của phần tử <p>.

Với phương thức addEventListener (), bạn có thể chỉ định loại lan truyền bằng cách sử dụng tham số "useCapture":

addEventListener(event, function, useCapture);

Giá trị mặc định là false, sẽ sử dụng lan truyền sủi bọt, khi giá trị được đặt thành true, sự kiện sử dụng lan truyền bắt giữ.

#### DOM Navigation

Theo tiêu chuẩn DOM W3C HTML, mọi thứ trong tài liệu HTML là một nút:

Toàn bộ tài liệu là một nút tài liệu

Mỗi phần tử HTML là một nút phần tử

Văn bản bên trong các phần tử HTML là các nút văn bản

Mỗi thuộc tính HTML là một nút thuộc tính (không dùng nữa)

Tất cả các bình luận là các nút bình luận

DOM HTML tree

Với HTML DOM, tất cả các nút trong cây nút có thể được truy cập bằng JavaScript.

Các nút mới có thể được tạo và tất cả các nút có thể được sửa đổi hoặc xóa.

Mối quan hệ nút

Các nút trong cây nút có mối quan hệ phân cấp với nhau.

Các thuật ngữ cha mẹ, con cái và anh chị em được sử dụng để mô tả các mối quan hệ.

Trong cây nút, nút trên cùng được gọi là nút gốc (hoặc nút gốc)

Mỗi nút có chính xác một cha, ngoại trừ gốc (không có cha)

Một nút có thể có một số con

Anh chị em (anh chị em) là các nút có cùng cha mẹ

Từ HTML ở trên, bạn có thể đọc:

<html> là nút gốc

<html> không có bố mẹ

<html>là cha mẹ của <head>và<body>

<head> là đứa con đầu lòng của <html>

<body> là đứa con cuối cùng của <html>

và:

<head> có một con: <title>

<title> có một con (một nút văn bản): "Hướng dẫn DOM"

<body>có hai con: <h1>và<p>

<h1> có một con: "DOM Bài học một"

<p> có một đứa con: "Xin chào thế giới!"

<h1>và <p>là anh chị em

Điều hướng giữa các nút

Bạn có thể sử dụng các thuộc tính nút sau để điều hướng giữa các nút bằng JavaScript:

* parentNode
* childNodes[nodenumber]
* firstChild
* lastChild
* nextSibling
* previousSibling
* Nút gốc DOM

Có hai thuộc tính đặc biệt cho phép truy cập vào tài liệu đầy đủ:

document.body - Phần thân của tài liệu

document.documentElement - Tài liệu đầy đủ

Thuộc tính nodeName

Các nodeNamebất động sản chỉ định tên của một nút.

* nodeName chỉ đọc
* nodeName của một nút phần tử giống như tên thẻ
* nodeName của một nút thuộc tính là tên thuộc tính
* nodeName của một nút văn bản luôn là #text
* nodeName của nút tài liệu luôn là #document

Thuộc tính nodeValue

Các nodeValuebất động sản xác định giá trị của một nút.

* nodeValue cho các nút phần tử là null
* nodeValue cho các nút văn bản là chính văn bản
* nodeValue cho các nút thuộc tính là giá trị thuộc tính

Thuộc tính nodeType

Các nodeTypetài sản chỉ được đọc. Nó trả về kiểu của một nút.

#### DOM Nodes

Tạo các phần tử HTML mới (Nút)

Để thêm một phần tử mới vào HTML DOM, trước tiên bạn phải tạo phần tử (nút phần tử), sau đó nối nó vào phần tử hiện có.

Thí dụ

<div id="div1">

<p id="p1">This is a paragraph.</p>

<p id="p2">This is another paragraph.</p>

</div>

<script>

var para = document.createElement("p");

var node = document.createTextNode("This is new.");

para.appendChild(node);

var element = document.getElementById("div1");

element.appendChild(para);

</script>

Giải thích ví dụ

Mã này tạo ra một <p>yếu tố mới :

var para = document.createElement("p");

Để thêm văn bản vào thành <p>phần, trước tiên bạn phải tạo một nút văn bản. Mã này tạo một nút văn bản:

var node = document.createTextNode("This is a new paragraph.");

Sau đó, bạn phải nối nút văn bản vào thành <p>phần:

para.appendChild(node);

Cuối cùng, bạn phải nối phần tử mới vào phần tử hiện có.

Mã này tìm thấy một yếu tố hiện có:

var element = document.getElementById("div1");

Mã này nối thêm phần tử mới vào phần tử hiện có:

element.appendChild(para);

Tạo các thành phần HTML mới - insertB Before ()

Thí dụ

<div id="div1">

<p id="p1">This is a paragraph.</p>

<p id="p2">This is another paragraph.</p>

</div>

<script>

var para = document.createElement("p");

var node = document.createTextNode("This is new.");

para.appendChild(node);

var element = document.getElementById("div1");

var child = document.getElementById("p1");

element.insertBefore(para, child);

</script>

Xóa các thành phần HTML hiện có

Để xóa phần tử HTML, hãy sử dụng remove():

Thí dụ

<div>

  <p id="p1">This is a paragraph.</p>

  <p id="p2">This is another paragraph.</p>

</div>

<script>

var elmnt = document.getElementById("p1");

elmnt.remove();

</script>

chứa một <div>phần tử có hai nút con (hai <p> phần tử):

<div>

  <p id="p1">This is a paragraph.</p>

  <p id="p2">This is another paragraph.</p>

</div>

Tìm phần tử bạn muốn xóa:

var elmnt = document.getElementById("p1");

Sau đó thực hiện phương thức remove () trên phần tử đó:

elmnt.remove();

Loại bỏ một nút con

Đối với các trình duyệt không hỗ trợ remove()phương thức, bạn phải tìm nút cha để xóa phần tử

Thí dụ

**<div** id="div1"**>**

**<p** id="p1"**>**This is a paragraph.**</p>**

**<p** id="p2"**>**This is another paragraph.**</p>**

**</div>**

**<script>**

var parent = document.getElementById("div1");

var child = document.getElementById("p1");

parent.removeChild(child);

**</script>**

#### DOM Collections

Đối tượng HTMLCollection

Các getElementsByTagName()phương thức trả về một HTMLCollectionđối tượng.

Một HTMLCollectionđối tượng là một danh sách giống như mảng (bộ sưu tập) các phần tử HTML.

Đoạn mã sau chọn tất cả các <p>thành phần trong tài liệu:

Thí dụ

var x = document.getElementsByTagName("p");

Các yếu tố trong bộ sưu tập có thể được truy cập bởi một số chỉ mục.

Để truy cập phần tử <p> thứ hai, bạn có thể viết:

y = x[1];

Độ dài HTMLCollection HTML

Các lengthbất động sản xác định số phần tử trong một HTMLCollection:

Thí dụ

var myCollection = document.getElementsByTagName("p");

document.getElementById("demo").innerHTML = myCollection.length;

Giải thích:

Tạo một bộ sưu tập của tất cả <p>các yếu tố

Hiển thị độ dài của bộ sưu tập

Các length rất hữu ích khi bạn muốn để lặp qua các yếu tố trong một bộ sưu tập:

Thay đổi màu nền của tất cả các phần tử <p>:

Vd:

var myCollection = document.getElementsByTagName("p");

var i;

for (i = 0; i **<** **myCollection.length**; i++) {

  myCollection[i].style.backgroundColor = "red";

}

#### DOM Node Lists

Đối tượng NodeList HTML DOM

Một NodeListđối tượng là một danh sách (tập hợp) các nút được trích xuất từ ​​một tài liệu.

Một NodeListđối tượng gần giống như một HTMLCollectionđối tượng.

Một số trình duyệt (cũ hơn) trả về một đối tượng NodeList thay vì HTMLCollection cho các phương thức như getElementsByClassName().

Tất cả các trình duyệt trả về một đối tượng NodeList cho thuộc tính childNodes.

Hầu hết các trình duyệt trả về một đối tượng NodeList cho phương thức querySelectorAll().

Đoạn mã sau chọn tất cả <p>các nút trong tài liệu:

Thí dụ

var myNodeList = document.querySelectorAll("p");

Các phần tử trong NodeList có thể được truy cập bằng số chỉ mục.

Để truy cập nút <p> thứ hai, bạn có thể viết:

y = myNodeList[1];

HTML DOM node list Length

Các length property xác định số nút trong một danh sách nút:

Thí dụ

var myNodelist = document.querySelectorAll("p");

document.getElementById("demo").innerHTML = myNodelist.length;

Giải thích:

Thí dụ

Thay đổi màu nền của tất cả các phần tử <p> trong danh sách nút:

var myNodelist = document.querySelectorAll("p");

var i;

for (i = 0; i **<** **myNodelist.length**; i++) {

  myNodelist[i].style.backgroundColor = "red";

}

### Mô hình đối tượng trình duyệt (BOM – Browser Object Model)

Không có một chuẩn chính thức nào cho tập hợp các mô hình đối tượng trình duyệt – BOM. Tuy nhiên các trình duyệt thường được trang bị các thuộc tính và hành vi tương đối giống nhau cho JavaScript sử dụng.

#### JS Window

Cửa sổ (window) là một đối tượng được hỗ trợ bởi tất cả các trình duyệt, nó thể hiện cửa sổ của trình duyệt. Toàn bộ các đối tượng toàn cục, hàm toàn cục và các biến toàn cục của JavaScript đều thuộc đối tượng window. Trong đó các biến toàn cục là các thuộc tính của đối tượng window. Các hàm toàn cục là các phương thức của đối tượng window và thậm trí các đối tượng tài liệu (HTML DOM) cũng là các thuộc tính của đối tượng window:

window.document.getElementById("header");

Kích thước bên trong của cửa sổ (đơn vị điểm ảnh - pixel) là phần hiện nội dung trang web được xác định thông qua thuộc tính innerHeight và innerWidth. Tuy nhiên cần lưu ý với trình duyệt cữ như Internet Explorer phiên bản < 8. Kích thước phần hiển thị trang web được xác định theo các cách như:

document.documentElement.clientHeight

document.documentElement.clientWidth

hoặc

document.body.clientHeight

document.body.clientWidth

Để chắc chắn có thể lấy được kích thước cửa sổ, thông thường sử dụng cách thức kết hợp như sau:

var w = window.innerWidth || document.documentElement.clientWidth || document.body.clientWidth;  
var h = window.innerHeight || document.documentElement.clientHeight || document.body.clientHeight;

Đối tượng cửa sổ cũng có các phương thức hữu ích thông dụng sau:

* window.open() – Mở một cửa sổ mới
* window.close() – Đóng cửa sổ hiện tại
* window.moveTo() – Di chuyển tới cửa sổ
* window.resizeTo() – Thay đổi kích thước cửa

#### JS Screen

Đối tượng Màn hình (screen) chứa các thông tin về màn hình của người dùng. Để sử dụng đối tượng này có thể viết window.screen hoặc screen. Một số thuộc tính của đối tượng này bao gồm:

* screen.width: chiều rộng màn hình
* screen.height: chiều cao màn hình
* screen.availWidth: chiều rộng khả dụng của màn hình
* screen.availHeight: chiều cao khả dụng của màn hình

Ví dụ để in kích thước màn hình của người dùng, có thể viết:

document.getElementById("demo").innerHTML = "Screen Width: " + screen.width;

document.getElementById("demo").innerHTML = "Screen Height: " + screen.height;

#### JS Location

Đối tượng window.location được sử dụng để lấy địa chỉ của web hiện tại hoặc điều hướng trình duyệt sang hiển thị một trang web khác. Một số thuộc tính thông dụng của đối tượng này bao gồm:

* window.location.href: trả về url của trang hiện tại
* window.location.hostname: trả về tên miền của trang hiện tại
* window.location.pathname: trả về đường dẫn và tên file của trang hiện thại
* window.location.protocol: trả về giao thức của trang hiện tại (ví dụ http, https)
* window.location.assign: tải một tài liệu

Ví dụ để biết url của trang hiện tại:

document.getElementById("demo").innerHTML = "Page location is " + window.location.href;

#### JS History

Đối tượng window.history chứa lịch sử duyệt trang của trình duyệt. Tuy nhiên để bảo vệ thông tin của người dùng, có một số giới hạn truy cập của đối tượng này. Đối tượng này cũng cung cấp

Đối window.historytượng có thể được viết mà không có tiền tố cửa sổ.

Để bảo vệ quyền riêng tư của người dùng, có những hạn chế về cách JavaScript có thể truy cập đối tượng này.

Một số phương pháp:

* history.back() - giống như nhấp lại vào trình duyệt
* history.forward() - giống như nhấp vào chuyển tiếp trong trình duyệt

#### JS Navigator

Cửa sổ điều hướng

Vài ví dụ:

navigator.appName

navigator.appCodeName

navigator.platform

Cookies trình duyệt

Các cookieEnabled tài sản trả về true nếu cookie được bật, nếu không sai:

* Tên ứng dụng trình duyệt
* Các appNametài sản trả về tên ứng dụng của trình duyệt
* Tên mã ứng dụng trình duyệt
* Các appCodeNametài sản trả về tên mã ứng dụng của trình duyệt
* Công cụ trình duyệt
* Các producttài sản trả về tên sản phẩm của động cơ trình duyệt
* Phiên bản trình duyệt
* Các appVersiontài sản trả về thông tin phiên bản về trình duyệt
* Tác nhân trình duyệt
* Các userAgenttài sản trả về header user-agent gửi bởi trình duyệt đến máy chủ
* Cảnh báo !!!

Thông tin từ đối tượng điều hướng thường có thể bị sai lệch và không nên được sử dụng để phát hiện các phiên bản trình duyệt vì:

* Các trình duyệt khác nhau có thể sử dụng cùng tên
* Dữ liệu điều hướng có thể được thay đổi bởi chủ sở hữu trình duyệt
* Một số trình duyệt xác định sai bản thân để bỏ qua kiểm tra trang web
* Trình duyệt không thể báo cáo hệ điều hành mới, được phát hành muộn hơn trình duyệt
* Nền tảng trình duyệt
* Các platformtài sản trả về nền tảng trình duyệt (hệ điều hành)
* Ngôn ngữ trình duyệt
* Các languagetài sản trả về ngôn ngữ của trình duyệt
* Java có được kích hoạt không?
* Các javaEnabled()phương pháp trả về true nếu Java được kích hoạt

#### JS Popup Alert

Hộp thông báo

Một hộp cảnh báo thường được sử dụng nếu bạn muốn đảm bảo thông tin đến người dùng.

Khi hộp thông báo bật lên, người dùng sẽ phải bấm "OK" để tiếp tục.

Cú pháp

window.alert("sometext");

Các window.alert()phương pháp có thể được viết không có tiền tố cửa sổ.

Thí dụ

alert("I am an alert box!");

Hộp xác nhận

Một hộp xác nhận thường được sử dụng nếu bạn muốn người dùng xác minh hoặc chấp nhận một cái gì đó.

Khi hộp xác nhận bật lên, người dùng sẽ phải nhấp vào "OK" hoặc "Hủy" để tiếp tục.

Nếu người dùng nhấp vào "OK", hộp sẽ trả về đúng . Nếu người dùng nhấp vào "Hủy", hộp sẽ trả về sai .

Cú pháp

window.confirm("sometext");

Các window.confirm()phương pháp có thể được viết không có tiền tố cửa sổ

Hộp nhắc nhở

Một hộp nhắc thường được sử dụng nếu bạn muốn người dùng nhập giá trị trước khi vào trang.

Khi hộp nhắc nhở bật lên, người dùng sẽ phải nhấp vào "OK" hoặc "Hủy" để tiếp tục sau khi nhập giá trị đầu vào.

Nếu người dùng nhấp vào "OK", hộp sẽ trả về giá trị đầu vào. Nếu người dùng nhấp vào "Hủy", hộp sẽ trả về null.

Cú pháp

window.prompt("sometext","defaultText");

Các window.prompt()phương pháp có thể được viết không có tiền tố cửa sổ.

Phá vỡ dòng

Để hiển thị ngắt dòng bên trong hộp bật lên, hãy sử dụng dấu gạch chéo ngược theo sau là ký tự n.

Thí dụ

alert("Hello\nHow are you?");

#### JS Timing

Sự kiện thời gian

Đối windowtượng cho phép thực thi mã theo các khoảng thời gian xác định.

Những khoảng thời gian này được gọi là sự kiện thời gian.

Hai phương thức chính để sử dụng với JavaScript là:

setTimeout(function, milliseconds)

Thực thi một chức năng, sau khi chờ một số mili giây xác định.

setInterval(function, milliseconds)

Giống như setTimeout (), nhưng lặp lại việc thực hiện hàm liên tục.

Các setTimeout()và setInterval()là cả hai phương pháp của đối tượng HTML DOM Window

Phương thức setTimeout ()

window.setTimeout(function, milliseconds);

Các window.setTimeout()phương pháp có thể được viết không có tiền tố cửa sổ.

Tham số đầu tiên là một hàm được thực thi.

Tham số thứ hai cho biết số mili giây trước khi thực hiện.

Làm thế nào để ngăn chặn việc thực thi?

Các clearTimeout()phương pháp dừng thực hiện các chức năng quy định tại setTimeout ().

window.clearTimeout(timeoutVariable)

Các window.clearTimeout()phương pháp có thể được viết không có tiền tố cửa sổ.

Các clearTimeout()phương pháp sử dụng biến trở về từ setTimeout():

myVar = setTimeout(function, milliseconds);

clearTimeout(myVar);

Nếu chức năng chưa được thực thi, bạn có thể dừng thực thi bằng cách gọi clearTimeout()

Phương thức setInterval ()

Các setInterval()phương pháp lặp đi lặp lại một chức năng nhất định ở mọi thời gian khoảng nhất định.

window.setInterval(function, milliseconds);

Các window.setInterval()phương pháp có thể được viết không có tiền tố cửa sổ.

Tham số đầu tiên là hàm được thực thi.

Tham số thứ hai cho biết độ dài của khoảng thời gian giữa mỗi lần thực hiện.

Ví dụ này thực thi một chức năng gọi là "myTimer" mỗi giây một lần (như đồng hồ kỹ thuật số).

Làm thế nào để ngăn chặn việc thực thi?

Các clearInterval()phương pháp dừng hành của hàm được xác định trong phương pháp setInterval ().

window.clearInterval(timerVariable)

Các window.clearInterval()phương pháp có thể được viết không có tiền tố cửa sổ.

Các clearInterval()phương pháp sử dụng biến trở về từ setInterval():

myVar = setInterval(function, milliseconds);

clearInterval(myVar);

#### JS Cookies

Cookies cho phép bạn lưu trữ thông tin người dùng trong các trang web.

Cookies là dữ liệu, được lưu trữ trong các tệp văn bản nhỏ, trên máy tính của bạn.

Khi một máy chủ web đã gửi một trang web tới trình duyệt, kết nối sẽ bị tắt và máy chủ sẽ quên mọi thứ về người dùng.

Cookies được phát minh để giải quyết vấn đề "làm thế nào để ghi nhớ thông tin về người dùng":

* Khi người dùng truy cập một trang web, tên của anh ấy / cô ấy có thể được lưu trữ trong một cookie.
* Lần tới khi người dùng truy cập trang, cookie sẽ "nhớ" tên của anh ấy / cô ấy.

Cookies được lưu trong các cặp tên-giá trị như: username = John Doe

Khi trình duyệt yêu cầu một trang web từ máy chủ, cookie thuộc về trang sẽ được thêm vào yêu cầu. Bằng cách này, máy chủ sẽ lấy dữ liệu cần thiết để "ghi nhớ" thông tin về người dùng.

Không có ví dụ nào dưới đây sẽ hoạt động nếu trình duyệt của bạn đã tắt hỗ trợ cookie cục bộ.

Tạo một cookie với JavaScript

JavaScript có thể tạo, đọc và xóa cookie với thuộc document.cookie tính.

Với JavaScript, cookie có thể được tạo như thế này:

document.cookie = "username=John Doe";

Bạn cũng có thể thêm ngày hết hạn (theo thời gian UTC). Theo mặc định, cookie sẽ bị xóa khi đóng trình duyệt:

document.cookie = "username=John Doe; expires=Thu, 18 Dec 2013 12:00:00 UTC";

Với tham số đường dẫn, bạn có thể cho trình duyệt biết cookie thuộc về đường dẫn nào. Theo mặc định, cookie thuộc về trang hiện tại.

document.cookie = "username=John Doe; expires=Thu, 18 Dec 2013 12:00:00 UTC; path=/";

Đọc một cookie với JavaScript

Với JavaScript, cookie có thể được đọc như thế này:

var x = document.cookie;

document.cookiesẽ trả về tất cả các cookie trong một chuỗi giống như: cookie1 = value; cookie2 = giá trị; cookie3 = giá trị;

Thay đổi Cookie bằng JavaScript

Với JavaScript, bạn có thể thay đổi cookie giống như cách bạn tạo:

document.cookie = "username=John Smith; expires=Thu, 18 Dec 2013 12:00:00 UTC; path=/";

Các cookie cũ được ghi đè.

Xóa cookie bằng JavaScript

Xóa một cookie rất đơn giản.

Bạn không phải chỉ định giá trị cookie khi xóa cookie.

Chỉ cần đặt tham số hết hạn thành ngày đã qua:

document.cookie = "username=; expires=Thu, 01 Jan 1970 00:00:00 UTC; path=/;";

Bạn nên xác định đường dẫn cookie để đảm bảo rằng bạn xóa đúng cookie.

Một số trình duyệt sẽ không cho phép bạn xóa cookie nếu bạn không chỉ định đường dẫn.

Chuỗi Cookie

Các document.cookietài sản trông giống như một chuỗi văn bản bình thường. Nhưng nó không phải như vậy.

Ngay cả khi bạn viết toàn bộ chuỗi cookie vào document.cookie, khi bạn đọc lại, bạn chỉ có thể thấy cặp tên-giá trị của nó.

Nếu bạn đặt cookie mới, cookie cũ sẽ không bị ghi đè. Cookie mới được thêm vào document.cookie, vì vậy nếu bạn đọc lại document.cookie, bạn sẽ nhận được một cái gì đó như:

cookie1 = giá trị; cookie2 = giá trị;

Hiển thị tất cả các cookie Tạo cookie 1 Tạo cookie 2 Xóa cookie 1 Xóa cookie 2

Nếu bạn muốn tìm giá trị của một cookie được chỉ định, bạn phải viết hàm JavaScript tìm kiếm giá trị cookie trong chuỗi cookie.

### Lập trình AJAX

#### AJAX Intro

AJAX là giấc mơ của một nhà phát triển, bởi vì bạn có thể :

- Đọc dữ liệu từ một máy chủ web – sau khi trang đã nạp.

- Cập nhật một trang web mà không cần tải lại trang.

- Gửi dữ liệu đến một máy chủ web – ở chế độ nền.

- Ví dụ và Giải thích :

<!DOCTYPE html>  
<html>  
<body>  
<div id="demo">  
  <h2>Let AJAX change this text</h2>  
  <button type="button" onclick="loadDoc()">Change Content</button>  
</div>  
</body>  
</html>

Trang HTML có chứa một thẻ <div> và thẻ <button>.

<div> được sử dụng để hiển thị thông tin từ một máy chủ.

<button> gọi đến một hàm (nếu nó được nhấp).

Hàm yêu cầu dữ liệu từ máy chủ và hiển thị chúng :

function loadDoc() {  
  var xhttp = new XMLHttpRequest();  
  xhttp.onreadystatechange = function() {  
    if (this.readyState == 4 && this.status == 200) {  
     document.getElementById("demo").innerHTML = this.responseText;  
    }  
  };  
  xhttp.open("GET", "ajax\_info.txt", true);  
  xhttp.send();  
}

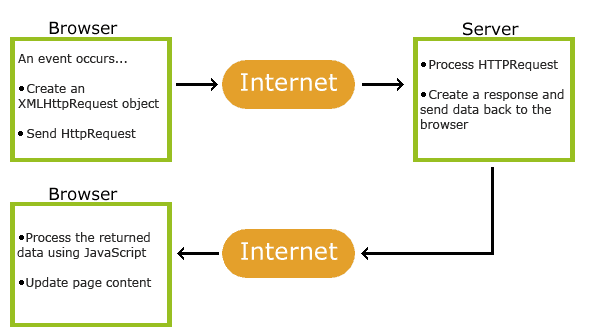
* AJAX là gì?

- AJAX là viết tắt cho Asynchronous JavaScript And XML.

- AJAX không phải là một ngôn ngữ lập trình.

- AJAX chỉ sử dụng sự kết hợp của :

* Một trình duyệt tích hợp trong đối tượng XMLHttpRequest (để yêu cầu dữ liệu từ một máy chủ web).
* JavaScript và HTML DOM (để hiển thị hoặc sử dụng các dữ liệu).
* AJAX là một cái tên dễ gây hiểu nhầm. Ứng dụng AJAX có thể sử dụng XML để vận chuyển dữ liệu, nhưng nó thường chuyển từ văn bản gốc sang văn bản JSON.
* AJAX cho phép các trang web cập nhật không đông bộ bằng cách trao đổi dữ liệu với một máy chủ web đằng sau hậu trường. Điều này có nghĩa rằng nó có thể cập nhật các phần của một trang web, mà không cần tải lại toàn bộ trang.
* AJAX làm việc như thế nào?



1. Một sự kiện xảy ra trong một trang web (trang web được tải, một nút được nhấp).

2. Một đối tượng XMLHttpRequest được tạo ra bởi JavaScript.

3. Đối tượng XMLHttpRequest gửi một yêu cầu đến một máy chủ web.

4. Máy chủ xử lý yêu cầu.

5. Máy chủ sẽ gửi một hồi đáp trở lại trang web.

6. Câu trả lời được đọc bởi JavaScript.

7. Hành động đúng ( như cập nhật trang) được thực hiện bởi JavaScript.

#### AJAX XMLHttp

Đối tượng XMLHttpRequest là viên đá đỉnh vòm của AJAX.

* Các đối tượng XMLHttpRequest
* Tất cả các trình duyệt hiện đại hỗ trợ các đối tượng XMLHttpRequest.
* Các đối tượng XMLHttpRequest có thể được sử dụng để trao đổi dữ liệu với một máy chủ web đằng sau màn hình. Điều này có nghĩa rằng nó có thể cập nhật các phần của một trang web, mà không cần tải lại toàn bộ trang.
* Tạo một đối tượng XMLHttpRequest
* Tất cả các trình duyệt hiện đại (Chrome, Firefox, IE7 +, Edge, Safari, Opera) có cài đặt sẵn đối tượng XMLHttpRequest.
* Cú pháp để tạo một đối tượng XMLHttpRequest :

*variable*= new XMLHttpRequest();

* Truy cập Across Domains
* Vì lý do an ninh, trình duyệt hiện đại không cho phép truy cập trên các miền.
* Điều này có nghĩa rằng cả trang web và các tập tin XML cố gắng để tải, phải nằm trên cùng một máy chủ.
* Nếu bạn muốn sử dụng ví dụ trên vào một trong các trang web của riêng bạn, các tập tin XML bạn nạp phải được đặt trên máy chủ của riêng bạn.
  + Các trình duyệt cũ (IE5 và IE6)
* Phiên bản cũ của trình duyệt Internet Explorer (5/6) sử dụng một đối tượng AtiveX thay cho đối tượng XMLHttpRequest :
* variable = new ActiveXObject (“Microsoft.XMLHTTP”);
* Để xử lý IE5 và IE6, kiểm tra nếu trình duyệt hỗ trợ các đối tượng XMLHttpRequest , nếu không tạo ra một đối tượng ActiveX
* Phương thức đối tượng XMLHttpRequest

|  |  |
| --- | --- |
| Phương thức | Mô tả |
| new XMLHttpRequest() | Tạo một đối tượng XMLHttpRequest mới |
| abort() | Hủy yêu cầu hiện tại |
| getAllResponseHeaders() | Trả về thông tin tiêu đề |
| getResponseHeader() | Trả về thông tin tiêu đề cụ thể. |
| open(method, url, async, user, psw) | Xác định yêu cầu :  method: yêu cầu kiểu GET hoặc POST  url: địa chỉ file  async: true (không đồng bộ) hoặc false (đồng bộ)  user: tên người dùng tùy chọn.  psw: mật khẩu tùy chọn. |
| send() | Gửi yêu cầu đến máy chủ.  sử dụng cho các yêu cầu GET. |
| send(string) | Gửi yêu cầu đến máy chủ.  sử dụng cho các yêu cầu POST. |
| setRequestHeader() | Thêm một cặp nhãn/giá trị vào tiêu đề sẽ được gửi. |

* Thuộc tính của đối tượng XMLHttpRequest

|  |  |
| --- | --- |
| Thuộc tính | Mô tả |
| onreadystatechange | Định nghĩa một hàm được gọi đến khi thuộc tính readyState thay đổi. |
| readyState | Giữ trạng thái của XMLHttpRequest  0: yêu cầu không được khởi tạo.  1: thiết lập kết nối máy chủ.  2: nhận yêu cầu.  3: xử lý yêu cầu  4: yêu cầu kết thúc và phản hồi sẵn sàng. |
| responseText | Trả về dữ liệu phản hồi dưới dạng chuỗi. |
| responseXML | Trả về dữ liệu phản hồi dưới dạng dữ liệu XML |
| status | Trả về số trạng thái của yêu cầu  200: "OK"  403: "Forbidden"  404: "Not Found" |
| statusText | Trả về trạng thái của văn bản (e.g. "OK" or "Not Found") |

#### AJAX Request

Đối tượng XMLHttprequest được sử dụng để trao đổi dữ liệu với một máy chủ.

* Gửi một yêu cầu đến máy chủ

Để gửi yêu cầu đến một máy chủ, ta dùng phương thức open() và send() của đối tượng XMLHttpRequest :

xhttp.open("GET", "ajax\_info.txt", true);  
xhttp.send();

|  |  |
| --- | --- |
| Phương thức | Mô tả |
| open(method, url, async) | Xác định kiểu của yêu cầu. method: kiểu của yêu cầu : GET or POST url: địa chỉ của máy chủ(file) async: true (không đồng bộ) or false (đồng bộ) |
| send() | Gửi yêu cầu đến máy chủ (dùng cho GET) |
| send(string) | Gửi yêu cầu đến máy chủ (dùng cho POST) |

* GET hoặc POST

GET đơn giản hơn và nhanh hơn POST, và có thể được sử dụng trong hầu hết các trường hợp.

Tuy nhiên, luôn sử dụng POST khi:

* Một tập tin được lưu trữ không phải là một tùy chọn (cập nhật một tập tin hoặc cơ sở dữ liệu trên máy chủ).
* Gửi một lượng lớn dữ liệu đến máy chủ (POST không có giới hạn kích thước).
* Gửi đầu vào người dùng (có thể chứa các ký tự lạ), POST mạnh hơn và an toàn hơn so với GET.
* Yêu cầu GET

Một yêu cầu GET đơn giản :

xhttp.open("GET", "demo\_get.asp", true);  
xhttp.send();

Trong ví dụ trên, bạn có thể có được một kết quả được lưu trữ. Để tránh điều này, thêm một ID suy nhất vào URL :

xhttp.open("GET", "demo\_get.asp?t=" + Math.random(), true);  
xhttp.send();

Nếu bạn muốn gửi thông tin với phương thức GET, thêm thông tin vào URL:

xhttp.open("GET", "demo\_get2.asp?fname=Henry&lname=Ford", true);  
xhttp.send();

* Yêu cầu POST

Một yêu cầu POST đơn giản :

xhttp.open("POST", "demo\_post.asp", true);  
xhttp.send();

Để dữ liệu POST như một HTML form, thêm một tiêu đề HTTP với setRequestHeader(). Xác định các dữ liệu mà bạn muốn gửi trong phương thức send() :

xhttp.open("POST", "ajax\_test.asp", true);  
xhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");  
xhttp.send("fname=Henry&lname=Ford");

|  |  |
| --- | --- |
| Phương thức | Mô tả |
| setRequestHeader(header, value) | Thêm tiêu đề HTTP cho yêu cầu  header: xác định tên tiêu đề  value: xác định giá trị của tiêu đề |

* URL – Một file trên máy chủ

Tham số URL của phương thức open() là một địa chỉ đến một tập tin trên máy chủ

xhttp.open("GET", "ajax\_test.asp", true);

Các tập tin có thể là bất kỳ loại tập tin, như .txt và .xml, hoặc các tập tin kịch bản máy chủ như asp và php (mà có thể thực hiện các thao tác trên máy chủ trước khi gửi phản hồi trở lại).

* Asynchronous – True or False?

Yêu cầu máy chủ phải được gửi đồng bộ.

Tham số async của phương thức open() nên được thiết lập là true :

xhttp.open("GET", "ajax\_test.asp", true);

Bằng cách gửi không đông bộ, JavaScript không nhất thiết chờ đợi phản hồi từ phía máy chủ, nhưng có thể thay vì :

* Thực hiện các Script khác trong khi chờ đợi phản hồi từ phía máy chủ.
* Đối phó với các phản hồi sau phản hồi đã sẵn sàng.
* Thuộc tính onreadystatechange

Với đối tượng XMLHttpRequest bạn có thể định nghĩa một hàm được thực thi khi yêu cầu nhận được một câu trả lời.

Hàm được định nghĩa trong thuộc tính onreadystatechange của đối tượng XMLHttpRequest :

xhttp.onreadystatechange = function() {  
  if (this.readyState == 4 && this.status == 200) {  
    document.getElementById("demo").innerHTML = this.responseText;  
  }  
};  
xhttp.open("GET", "ajax\_info.txt", true);  
xhttp.send();

* Yêu cầu đồng bộ

Để thực hiện một yêu cầu đồng bộ, thay đổi tham số thứ ba trong phương thức open() là false:



Đôi khi async=false được sử dụng để kiểm tra nhanh chóng. Bạn cũng sẽ tìm thấy yêu cầu đồng bộ trong mã JavaScript cũ.

Khi mã chờ đợi sự hoàn thành phía máy chủ, không cần có một hàm onreadystatechange :

xhttp.open("GET", "ajax\_info.txt", false);  
xhttp.send();  
document.getElementById("demo").innerHTML = xhttp.responseText;

Chú ý :

* XMLHttpRequest đồng bộ (async=false) là không nên vì JavaScript sẽ ngừng thực hiện cho đến khi phản hồi phía máy chủ là đã sẵn sang. Nếu máy chủ đang bận hoặc chậm, ứng dụng sẽ treo hoặc dừng lại.
* XMLHttpRequest trong quá trình đang được gỡ bỏ từ các tiêu chuẩn web, nhưng quá trình này có thể mất nhiều năm.
* Công cụ phát triển hiện đại được khuyến khích để cảnh báo về việc sử dụng yêu cầu đồng bộ và có thể nén một ngoại lệ InvalidAccessError khi nó xảy ra.

#### AJAX Response

* Thuộc tính onreadystatechange

Các thuộc tính readyState nắm trạng thái của XMLHttpRequest.

Các thuộc tính onreadystatechange định nghĩa một hàm được thực thi khi readyState thay đổi.

Các thuộc tính status và các thuộc tính statusText giữ trạng thái của đối tượng XMLHttpRequest.

|  |  |
| --- | --- |
| Thuộc tính | Miêu tả |
| onreadystatechange | Định nghĩa hàm được gọi khi thuộc tính readyStae thay đổi. |
| readyState | Giữ trạng thái của XMLHttpRequest.  0: yêu cầu không được khởi tạo  1: kết nối máy chủ được thiết lập  2: yêu cầu được nhận  3: xử lý yêu cầu  4: kết thúc yêu cầu và phản hồi sẵn sàng |
| status | 200: "OK"  403: "Forbidden"  404: "Page not found" |
| statusText | Trả lại tráng thái văn bản (e.g. "OK" or "Not Found") |

Các hàm *onreadystatechange* được gọi mỗi khi thay đổi readyState.

Khi readyState bằng 4 và status bằng 200, phản hồi đã sẵn sàng :

function loadDoc() {  
  var xhttp = new XMLHttpRequest();  
  xhttp.onreadystatechange = function() {  
    if (this.readyState == 4 && this.status == 200) {  
      document.getElementById("demo").innerHTML =  
      this.responseText;  
    }  
  };  
  xhttp.open("GET", "ajax\_info.txt", true);  
  xhttp.send();  
}

Các sự kiện onreadystatechange được kích hoạt 4 lần (1-4), ứng cho mỗi sự thay đổi trong readyState.

* Sử dụng một hàm Callback

Một hàm callback là một hàm thông qua như là một tham số tới một hàm khác.

Nếu bạn có nhiều hơn một nhiệm vụ AJAX trong một trang web, bạn nên tạo một hàm để thực hiện các đối tượng XMLHttpRequest, và một hàm gọi lại cho mỗi công việc AJAX.

Các hàm callback nên chứa URL và những hàm để gọi khi phản hồi đã sẵn sàng.

loadDoc("url-1", myFunction1);  
loadDoc("url-2", myFunction2);  
function loadDoc(url, cFunction) {  
  var xhttp;  
  xhttp = new XMLHttpRequest();  
  xhttp.onreadystatechange = function() {  
    if (this.readyState == 4 && this.status == 200) {  
      cFunction(this);  
    }  
  };  
  xhttp.open("GET", url, true);  
  xhttp.send();  
}  
function myFunction1(xhttp) {  
  // action goes here  
}  
function myFunction2(xhttp) {  
  // action goes here  
}

* Thuộc tính Server Response

|  |  |
| --- | --- |
| Thuộc tính | Mô tả |
| responseText | Lấy dữ liệu phản hồi dưới dạng chuỗi |
| responseXML | Lấy dữ liệu phản hồi dưới dạng dữ liệu XML |

* Phương thức Server Response

|  |  |
| --- | --- |
| Phương thức | Mô tả |
| getResponseHeader() | Trả về thông tin tiêu đề cụ thể từ tài nguyên máy chủ. |
| getAllResponseHeaders() | Trả về tất cả thông tin tiêu đề từ tài nguyên máy chủ. |

* Thuộc tính responseText

Thuộc tính responseText trả về phản hồi máy chủ dưới dạng một chuỗi JavaScript, và bạn có thể sử dụng nó cho phù hợp :

document.getElementById("demo").innerHTML = xhttp.responseText;

* Thuộc tính responseXML

Đối tượng *XMLHttpRequest* có trình phân tích cú pháp XML dựng sẵn.

Thuộc tính responseXML trả về phản hồi của máy chủ dưới dạng đối tượng XML DOM.

Sử dụng thuộc tính này, bạn có thể phân tích phản hồi dưới dạng đối tượng XML DOM:

xmlDoc = xhttp.responseXML;  
txt = "";  
x = xmlDoc.getElementsByTagName("ARTIST");  
for (i = 0; i < x.length; i++) {  
  txt += x[i].childNodes[0].nodeValue + "<br>";  
}  
document.getElementById("demo").innerHTML = txt;  
xhttp.open("GET", "cd\_catalog.xml", true);  
xhttp.send();

* Phương thức getAllResponseHeaders()

Phương thức *getAllResponseHeaders()* trả về tất cả thông tin tiêu đề từ phản hồi của máy chủ.

var xhttp = new XMLHttpRequest();  
xhttp.onreadystatechange = function() {  
  if (this.readyState == 4 && this.status == 200) {  
    document.getElementById("demo").innerHTML =  
    this.getAllResponseHeaders();  
  }  
};

* Phương thức getResponseHeader()

Phương thức *getResponseHeader()* trả về thông tin tiêu đề cụ thể từ phản hồi của máy chủ.

var xhttp = new XMLHttpRequest();  
xhttp.onreadystatechange = function() {  
  if (this.readyState == 4 && this.status == 200) {  
    document.getElementById("demo").innerHTML =  
    this.getResponseHeader("Last-Modified");  
  }  
};  
xhttp.open("GET", "ajax\_info.txt", true);  
xhttp.send();

#### AJAX XML File

AJAX có thể được sử dụng để giao tiếp tương tác với một tập tin XML.

* Ví dụ AJAX XML



* Giải thích

Khi người dùng nhấp vào nút “Get info CD” ở trên, hàm loadDoc() được thực thi.

Hàm loadDoc() tạo ra một đối tượng XMLHttpRequest, thêm các hàm được thực thi khi các phản hồi từ máy chủ đã sẵn sàng, và gửi yêu cầu tắt đến máy chủ.

Khi phản hồi máy chủ đã sẵn sàng, một bảng HTML được xây dựng, các nút (phần tử) được chiết xuất từ các tập tin XML, và cuối cùng cập nhật các phần tử “demo” với bảng HTML đầy dữ liệu XML :

LoadXMLDoc()

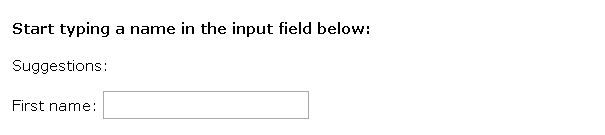
function loadDoc() {  
  var xhttp = new XMLHttpRequest();  
  xhttp.onreadystatechange = function() {  
    if (this.readyState == 4 && this.status == 200) {  
    myFunction(this);  
    }  
  };  
  xhttp.open("GET", "cd\_catalog.xml", true);  
  xhttp.send();  
}  
function myFunction(xml) {  
  var i;  
  var xmlDoc = xml.responseXML;  
  var table="<tr><th>Artist</th><th>Title</th></tr>";  
  var x = xmlDoc.getElementsByTagName("CD");  
  for (i = 0; i <x.length; i++) {  
    table += "<tr><td>" +  
    x[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +  
    "</td><td>" +  
    x[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue +  
    "</td></tr>";  
  }  
  document.getElementById("demo").innerHTML = table;  
}

#### AJAX PHP

AJAX được sử dụng để tạo ra các ứng dụng tương tác nhiều hơn.

* Ví dụ AJAX PHP

Ví dụ sau đây cho thấy làm thế nào một trang web có thể liên lạc với một máy chủ web trong khi người dùng nhập các ký tự trong trường nhập.



* Giải thích

Trong ví dụ trên, khi người dùng nhập một ký tự trong trường đầu vào, một hàm showHint() sẽ được thực thi.

Hàm này được kích hoạt bởi sự kiện *onkeyup*.

Đây là phần HTML code :

<html>  
<head>  
<script>  
function showHint(str) {  
  if (str.length == 0) {  
    document.getElementById("txtHint").innerHTML = "";  
    return;  
  } else {  
    var xmlhttp = new XMLHttpRequest();  
    xmlhttp.onreadystatechange = function() {  
      if (this.readyState == 4 && this.status == 200) {  
        document.getElementById("txtHint").innerHTML = this.responseText;  
      }  
    };  
    xmlhttp.open("GET", "gethint.php?q=" + str, true);  
    xmlhttp.send();  
  }  
}  
</script>  
</head>  
<body>  
<p><b>Start typing a name in the input field below:</b></p>  
<form>  
First name: <input type="text" onkeyup="showHint(this.value)">  
</form>  
<p>Suggestions: <span id="txtHint"></span></p>  
</body>  
</html>

Giải thích :

Đầu tiên, kiểm tra xem trường đầu vào có rỗng (str.lenghth ==0). Nếu có, xóa toàn bộ nội dung của placeholder txtHint và thoát khỏi hàm.

Tuy nhiên, nếu trường đầu vào không rỗng, làm như sau :

* Tạo một đối tượng XMLHttpRequest
* Tạo hàm được thực hiện khi phản hồi máy chủ đã sẵn sàng.
* Gửi yêu cầu tắt vào một tập tin PHP (fethint.php) trên máy chủ.
* Chú ý rằng tham số q được thêm gethint.php?q=”+str
* Biến str chứa dữ liệu của các trường đầu vào.
* PHP file – “gethint.php”

Tập tin PHP kiểm tra một loạt các tên và trả về tên tương ứng cho trình duyệt.

<?php  
// Array with names  
$a[] = "Anna";  
$a[] = "Brittany";  
$a[] = "Cinderella";  
$a[] = "Diana";  
$a[] = "Eva";  
$a[] = "Fiona";  
$a[] = "Gunda";  
$a[] = "Hege";  
$a[] = "Inga";  
$a[] = "Johanna";  
$a[] = "Kitty";  
$a[] = "Linda";  
$a[] = "Nina";  
$a[] = "Ophelia";  
$a[] = "Petunia";  
$a[] = "Amanda";  
$a[] = "Raquel";  
$a[] = "Cindy";  
$a[] = "Doris";  
$a[] = "Eve";  
$a[] = "Evita";  
$a[] = "Sunniva";  
$a[] = "Tove";  
$a[] = "Unni";  
$a[] = "Violet";  
$a[] = "Liza";  
$a[] = "Elizabeth";  
$a[] = "Ellen";  
$a[] = "Wenche";  
$a[] = "Vicky";  
// get the q parameter from URL  
$q = $\_REQUEST["q"];  
$hint = "";  
// lookup all hints from array if $q is different from ""  
if ($q !== "") {  
  $q = strtolower($q);  
  $len=strlen($q);  
  foreach($a as $name) {  
    if (stristr($q, substr($name, 0, $len))) {  
      if ($hint === "") {  
        $hint = $name;  
      } else {  
        $hint .= ", $name";  
      }  
    }  
  }  
}  
// Output "no suggestion" if no hint was found or output correct values  
echo $hint === "" ? "no suggestion" : $hint;  
?>

#### AJAX ASP

* Ví dụ AJAX ASP

Ví dụ sau đây cho thấy làm thế nào một trang web có thể liên lạc với một máy chủ web trong khi người dùng nhập các ký tự trong trường nhập.



* Giải thích

Trong ví dụ trên, khi người dùng nhập một ký tự trong trường đầu vào, một hàm showHint() được thực thi.

Hàm này được kích hoạt bởi các sự kiện onkeyup.

Đây là đoạn mã HTML:

<html>  
<head>  
<script>  
function showHint(str) {  
  if (str.length == 0) {  
    document.getElementById("txtHint").innerHTML = "";  
    return;  
  } else {  
    var xmlhttp = new XMLHttpRequest();  
    xmlhttp.onreadystatechange = function() {  
      if (this.readyState == 4 && this.status == 200) {  
        document.getElementById("txtHint").innerHTML = this.responseText;  
      }  
    };  
    xmlhttp.open("GET", "gethint.asp?q=" + str, true);  
    xmlhttp.send();  
  }  
}  
</script>  
</head>  
<body>  
  
<p><b>Start typing a name in the input field below:</b></p>  
<form>  
First name: <input type="text" onkeyup="showHint(this.value)">  
</form>  
<p>Suggestions: <span id="txtHint"></span></p>  
</body>  
</html>

* Giải thích :

Đầu tiên, kiểm tra xem trường đầu vào có rỗng (str.lenghth ==0). Nếu có, xóa toàn bộ nội dung của placeholder txtHint và thoát khỏi hàm.

Tuy nhiên, nếu trường đầu vào không rỗng, làm như sau :

* Tạo một đối tượng XMLHttpRequest
* Tạo hàm được thực hiện khi phản hồi máy chủ đã sẵn sàng.
* Gửi yêu cầu tắt vào một tập tin PHP (fethint.php) trên máy chủ.
* Chú ý rằng tham số q được thêm gethint.php?q=”+str
* Biến str chứa dữ liệu của các trường đầu vào.
* ASP file – “gethint.asp”

Các tập tin ASP kiểm tra một loạt các tên, và trả về tên tương ứng cho trình duyệt :

<%  
response.expires=-1  
dim a(30)  
'Fill up array with names  
a(1)="Anna"  
a(2)="Brittany"  
a(3)="Cinderella"  
a(4)="Diana"  
a(5)="Eva"  
a(6)="Fiona"  
a(7)="Gunda"  
a(8)="Hege"  
a(9)="Inga"  
a(10)="Johanna"  
a(11)="Kitty"  
a(12)="Linda"  
a(13)="Nina"  
a(14)="Ophelia"  
a(15)="Petunia"  
a(16)="Amanda"  
a(17)="Raquel"  
a(18)="Cindy"  
a(19)="Doris"  
a(20)="Eve"  
a(21)="Evita"  
a(22)="Sunniva"  
a(23)="Tove"  
a(24)="Unni"  
a(25)="Violet"  
a(26)="Liza"  
a(27)="Elizabeth"  
a(28)="Ellen"  
a(29)="Wenche"  
a(30)="Vicky"  
'get the q parameter from URL  
q=ucase(request.querystring("q"))  
'lookup all hints from array if length of q>0  
if len(q)>0 then  
  hint=""  
  for i=1 to 30  
    if q=ucase(mid(a(i),1,len(q))) then  
      if hint="" then  
        hint=a(i)  
      else  
        hint=hint & " , " & a(i)  
      end if  
    end if  
  next  
end if  
'Output "no suggestion" if no hint were found  
'or output the correct values  
if hint="" then  
  response.write("no suggestion")  
else  
  response.write(hint)  
end if  
%>

#### AJAX Database

AJAX có thể được sử dụng để giao tiếp tương tác với một cơ sở dữ liệu.

* Ví dụ cơ sở dũ liệu AJAX

Ví dụ sau đây sẽ chứng minh làm thế nào một trang web có thể lấy thông tin từ một cơ sở dữ liệu với AJAX.



* Giải thích – hàm showCustomer()

Khi người dùng chọn một khách hàng trong danh sách thả xuống ở trên, một hàm showCustomer() được thực thi. Chức năng này được kích hoạt bởi các sự kiên *onchange* :

showCustomer

function showCustomer(str) {  
  var xhttp;  
  if (str == "") {  
    document.getElementById("txtHint").innerHTML = "";  
    return;  
  }  
  xhttp = new XMLHttpRequest();  
  xhttp.onreadystatechange = function() {  
    if (this.readyState == 4 && this.status == 200) {  
    document.getElementById("txtHint").innerHTML = this.responseText;  
    }  
  };  
  xhttp.open("GET", "getcustomer.php?q="+str, true);  
  xhttp.send();  
}

Chức năng của showCustomer() :

* Kiểm tra nếu một khách hàng được chọn.
* Tạo một đối tượng XMLHttpRequest.
* Tạo hàm được thực thi khi các phản hồi máy chủ đã sẵn sàng.
* Gửi yêu cầu tắt vào một tập tin trên máy chủ.
* Chú ý rằng một tham số q sẽ được thêm vào URL (với nội dung của danh sách thả xuống).
* AJAX Server Page

Các trang trên máy chủ được gọi bởi JavaScript ở trên là một file PHP gọi là “getcustomer.php”.

Các mã nguồn trong “getcustomer.php” chạy một truy vấn đối với một cơ sở dữ liệu, và trả về kết quả trong một bảng HTML :

<?php  
$mysqli = new mysqli("servername", "username", "password", "dbname");  
if($mysqli->connect\_error) {  
  exit('Could not connect');  
}  
$sql = "SELECT customerid, companyname, contactname, address, city, postalcode, country  
FROM customers WHERE customerid = ?";  
$stmt = $mysqli->prepare($sql);  
$stmt->bind\_param("s", $\_GET['q']);  
$stmt->execute();  
$stmt->store\_result();  
$stmt->bind\_result($cid, $cname, $name, $adr, $city, $pcode, $country);  
$stmt->fetch();  
$stmt->close();  
echo "<table>";  
echo "<tr>";  
echo "<th>CustomerID</th>";  
echo "<td>" . $cid . "</td>";  
echo "<th>CompanyName</th>";  
echo "<td>" . $cname . "</td>";  
echo "<th>ContactName</th>";  
echo "<td>" . $name . "</td>";  
echo "<th>Address</th>";  
echo "<td>" . $adr . "</td>";  
echo "<th>City</th>";  
echo "<td>" . $city . "</td>";  
echo "<th>PostalCode</th>";  
echo "<td>" . $pcode . "</td>";  
echo "<th>Country</th>";  
echo "<td>" . $country . "</td>";  
echo "</tr>";  
echo "</table>";  
?>

#### AJAX Applications

* Các tài liệu XML được sử dụng

Chúng ta sẽ sử dụng các tập tin XML được gọi là “cd catalog.xml” .

* Dữ liệu XML hiển thị trong một bảng HTML

Ví dụ này lặp qua mỗi phần tử <CD>, và hiển thị các giá trị của phần tử <ARTIST> và <TITLE> trong một bảng HTML :

<html>  
<head>  
<style>  
table, th, td {  
  border: 1px solid black;  
  border-collapse: collapse;  
}  
th, td {  
  padding: 5px;  
}  
</style>  
</head>  
<body>  
<table id="demo"></table>  
<script>  
function loadXMLDoc() {  
  var xmlhttp = new XMLHttpRequest();  
  xmlhttp.onreadystatechange = function() {  
    if (this.readyState == 4 && this.status == 200) {  
      myFunction(this);  
    }  
  };  
  xmlhttp.open("GET", "cd\_catalog.xml", true);  
  xmlhttp.send();  
}  
function myFunction(xml) {  
  var i;  
  var xmlDoc = xml.responseXML;  
  var table="<tr><th>Artist</th><th>Title</th></tr>";  
  var x = xmlDoc.getElementsByTagName("CD");  
  for (i = 0; i <x.length; i++) {  
    table += "<tr><td>" +  
    x[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +  
    "</td><td>" +  
    x[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue +  
    "</td></tr>";  
  }  
  document.getElementById("demo").innerHTML = table;  
}  
</script>  
</body>  
</html>

* Hiển thị CD đầu tiên trong một phần tử div HTML

Ví dụ này sử dụng một hàm để hiển thị các yếu tố CD đầu tiên trong một phần tử HTML với id= “showCD” .

displayCD(0);  
function displayCD(i) {  
  var xmlhttp = new XMLHttpRequest();  
  xmlhttp.onreadystatechange = function() {  
    if (this.readyState == 4 && this.status == 200) {  
      myFunction(this, i);  
    }  
  };  
  xmlhttp.open("GET", "cd\_catalog.xml", true);  
  xmlhttp.send();  
}  
function myFunction(xml, i) {  
  var xmlDoc = xml.responseXML;  
  x = xmlDoc.getElementsByTagName("CD");  
  document.getElementById("showCD").innerHTML =  
  "Artist: " +  
  x[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +  
  "<br>Title: " +  
  x[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue +  
  "<br>Year: " +  
  x[i].getElementsByTagName("YEAR")[0].childNodes[0].nodeValue;  
}

* Điều hướng giữa các đĩa CD

Để di chuyển giữa các đĩa CD trong ví dụ trên, tạo hàm next() và previous() :

function next() {  
  // display the next CD, unless you are on the last CD  
  if (i < len-1) {  
    i++;  
    displayCD(i);  
  }  
}  
  
function previous() {  
  // display the previous CD, unless you are on the first CD  
  if (i > 0) {  
    i--;  
    displayCD(i);  
  }  
}

* Hiện thông tin Album khi nhấp vào một đĩa CD

Ví dụ cuối cùng cho thấy làm thế nào bạn có thể hiển thị thông tin album khi người dùng nhấp vào một đĩa CD :

function displayCD(i) {  
  document.getElementById("showCD").innerHTML =  
  "Artist: " +  
  x[i].getElementsByTagName("ARTIST")[0].childNodes[0].nodeValue +  
  "<br>Title: " +  
  x[i].getElementsByTagName("TITLE")[0].childNodes[0].nodeValue +  
  "<br>Year: " +  
  x[i].getElementsByTagName("YEAR")[0].childNodes[0].nodeValue;  
}

### Sử dụng JSON

#### JSON Intro

JSON viết tắt của Java Script Object Notation.

JSON là một cú pháp để lưu trữ và trao đổi dữ liệu.

JSON là văn bản, được viết với ký hiệu đối tượng JavaScript.

* Trao đổi dữ liệu

Khi trao đổi giữa trình duyệt và máy chủ, dữ liệu chỉ có thể là văn bản.

JSON là văn bản, và chúng ta có thể chuyển đổi bất kỳ đối tượng JavaScript vào JSON, và gửi JSON lên máy chủ.

Chúng ta cũng có thể chuyển đổi bất kỳ JSON nhận từ máy chủ thành các đối tượng JavaScript.

Bằng cách này chúng ta có thể làm việc với các dữ liệu như đối tượng JavaScript, không có phân tích cú pháp phức tạp và bản dịch.

* Gửi dữ liệu

Nếu bạn có dữ liệu được lưu trữ trong một đối tượng JavaScript, bạn có thể chuyển đổi các đối tượng vào JSON, và gửi nó đến máy chủ:

var myObj = {name: "John", age: 31, city: "New York"};  
var myJSON = JSON.stringify(myObj);  
window.location = "demo\_json.php?x=" + myJSON;

* Nhận dữ liệu

Nếu bạn nhận dữ liệu ở định dạng JSON, bạn có thể chuyển đổi nó thành một đối tượng JavaScript.

var myJSON = '{"name":"John", "age":31, "city":"New York"}';  
var myObj = JSON.parse(myJSON);  
document.getElementById("demo").innerHTML = myObj.name;

* Lưu trữ dữ liệu

Khi lưu trữ dữ liệu, dữ liệu phải là một định dạng nhất định, và bất kể nơi bạn chọn để lưu trữ nó, văn bản luôn là một trong những định dạng quy phạm pháp luật.

JSON làm cho nó có thể lưu trữ các đối tượng JavaScript dưới sạng văn bản.

// Storing data:  
myObj = {name: "John", age: 31, city: "New York"};  
myJSON = JSON.stringify(myObj);  
localStorage.setItem("testJSON", myJSON);  
  
// Retrieving data:  
text = localStorage.getItem("testJSON");  
obj = JSON.parse(text);  
document.getElementById("demo").innerHTML = obj.name;

* JSON là gì?
* JSON là viết tắt của Java Script Object Notation.
* JSON là một định dạng dữ liệu trao đổi nhẹ.
* JSON là “tự mô tả” và dễ hiểu.
* JSON là ngôn ngữ độc lập (JSON sử dụng cú pháp JavaScript nhưng định dạng Json chỉ là văn bản. Văn bản có thể được đọc và sử dụng như là một định dạng dữ liệu bằng bất kỳ ngôn ngữ lập trình).
* Định dạng JSON ban đầu được xác định bởi Douglas Crockford.
* Tại sao lại sử dụng JSON?

Kể từ khi định dạng JSON chỉ là văn bản, nó có thể dễ dàng được gửi đến và đi từ một máy chủ, và được sử dụng như một định dạng dữ liệu bằng bất kỳ ngôn ngữ lập trình nào.

JavaScript đã tích hợp sẵn chức năng để chuyển đổi một chuỗi, viết bằng định dạng JSON , thành đối tượng JavaScript mẹ đẻ :

JSON.parse()

Vì vậy nếu bạn nhận được dữ liệu từ một máy chủ ở định dạng JSON, bạn có thể sử dụng nó như bất kỳ đối tượng JavaScript khác.

#### JSON Syntax

Cú pháp JSON là một tập hợp con của cú pháp JavaScript.

* Nội quy cú pháp JSON

Cú pháp JSON có nguồn gốc từ cú pháp ký hiệu đối tượng JavaScript :

* Dữ liệu trong tên/ cặp giá trị
* Dữ liệu được phân cách bằng dấu phẩy
* Dấu ngoặc nhọn giữ các đối tượng
* Dấu ngoặc vuông giữ mảng.
* JSON dữ liệu – một tên và một giá trị

Dữ liệu JSON được viết dưới dạng cặp tên/ giá trị.

Một cặp tên/ giá trị bao gồm một tên trường (trong dấu ngoặc kép), tiếp theo là một dấu hai chấm, tiếp theo là một giá trị :

Ví dụ :

"name":"John"

Chú ý : tên Json cần dấu ngoặc kép, tên JavaScript thì không.

* JSON – đánh giá đối tượng JavaScript

Định dạng JSON gần như giống hết với các đối tượng JavaScript.

Trong JSON, từ khóa phải có dạng chuỗi, được viết với dấu ngoặc kép :

JSON

{ "name":"John" }

Trong JavaScript, từ khóa có thể là chuỗi, số, hoặc tên định danh :

JavaScript

{ name:"John" }

* JSON value

Trong JSON, value phải là một trong các kiểu dữ liệu sau:

* Một chuỗi
* Một số
* Một đối tượng (đối tượng JSON)
* Một mảng
* Một boolean
* Rỗng

Trong JavaScript value có thể là tất cả các kiểu trên, cộng với bất kỳ biểu hiện JavaScript hợp lệ khác, bao gồm :

* Một hàm
* Một ngày
* Chưa xác định

Trong JSON, giá trị chuỗi phải được viết với dấu ngoặc kép:

{ "name":"John" }

Trong JavaScript, bạn có thể viết các giá trị chuỗi với đôi hoặc dấu nháy đơn :

{ name:'John' }

* JSON sử dụng cú pháp JavaScript

Bởi vì cú pháp JSON có nguồn gốc từ ký hiệu đối tượng JavaScript, rất ít phần mềm thêm là cần thiết để làm việc với JSON trong JavaScript.

Với JavaScript bạn có thể tạo một đối tượng và gán dữ liệu cho nó, như sau :

var person = { name: "John", age: 31, city: "New York" };

Bạn có thể truy nhập vào đối tượng JavaScript như sau:

// returns John  
person.name;

Nó cũng có thể được truy nhập như thế này :

// returns John  
person["name"];

Dữ liệu có thể được sửa đổi như sau :

person.name = "Gilbert";

Hoặc :

person["name"] = "Gilbert";

* Mảng JavaScript

Cũng giống như cách đối tượng JavaScript có thể sử dụng như mảng JSON, JavaScript cũng có thể được sử dụng như JSON.

* Tập tin JSON

Loại của các tập tin JSON là “.json”

Loại MIME cho JSON văn bản là “application/json”

#### JSON vs XML

Cả JSON và XML đều có thể được sử dụng để nhận dữ liệu từ máy chủ web.

Ví dụ dưới đây JSON và XML cùng xác định đối tượng một nhân viên, với một mảng gồm 3 nhân viên :

JSON Example

{"employees":[  
  { "firstName":"John", "lastName":"Doe" },  
  { "firstName":"Anna", "lastName":"Smith" },  
  { "firstName":"Peter", "lastName":"Jones" }  
]}

XML Example

<employees>  
  <employee>  
    <firstName>John</firstName> <lastName>Doe</lastName>  
  </employee>  
  <employee>  
    <firstName>Anna</firstName> <lastName>Smith</lastName>  
  </employee>  
  <employee>  
    <firstName>Peter</firstName> <lastName>Jones</lastName>  
  </employee>  
</employees>

* JSON giống XML vì
* Cả JSON và XML đều là “tự mô tả” (human readable)
* Cả hai JSON và XML là phân cấp (giá trị trong pham vi giá trị )
* Cả hai JSON và XML có thể được phân tích và được sử dụng bởi rất nhiều ngôn ngữ lập trình.
* Cả hai JSON và XML có thể được lấy với một XMLHttpRequest .
* JSON không giống XML vì
* JSON không sử dụng thẻ kết thúc
* JSON ngắn hơn
* JSON đọc và ghi nhanh hơn
* JSON có thể sử dụng mảng
* Tại sao JSON tốt hơn XML

XML khó phân tích hơn JSON.

JSON được phân tách thành một đối tượng JavaScript sẵn sàng để sử dụng.

Đối với các ứng dụng AJAX, JSON nhanh hơn và dễ dàng hơn XML.

Sử dụng XML :

* Fetch một tài liệu XML
* Sử dụng DOM XML để lặp qua các tài liệu.
* Giá trị giải nén và lưu trữ trong các biến.

Sử dụng JSON :

* Fetch một chuỗi JSON
* JSON.Parse chuỗi JSON

#### JSON Data Types

* Kiểu dữ liệu

Trong JSON, giá trị phải là một trong các kiểu dữ liệu sau :

* Một chuỗi
* Một số
* Một đối tượng (đối tượng JSON)
* Một mảng
* Một boolean
* Rỗng

Giá trị JSON không thể là một trong những kiểu dữ liệu sau :

* Một hàm
* Một ngày
* Chưa xác định
* JSON Strings

Strings trong JSON phải được viết trong dấu ngoặc kép.

{ "name":"John" }

* JSON number

Số trong JSON phải là số nguyên hoặc một dấu chấm động.

{ "age":30 }

* JSON object

Giá trị trong JSON có thể là các đối tượng.

{  
"employee":{ "name":"John", "age":30, "city":"New York" }  
}

Đối tượng như các giá trị trong JSON phải tuân theo các quy tắc tương tự như các đối tượng JSON.

* JSON Array

Giá trị trong JSON có thể là mảng.

{  
"employees":[ "John", "Anna", "Peter" ]  
}

* JSON Boolean

Giá trị trong JSON có thể đúng/sai.

{ "sale":true }

* JSON Null

Giá trị trong JSON có thể rỗng.

{ "middlename":null }

#### JSON Parse

JSON thường được dùng để trao đổi dữ liệu đến/ từ một máy chủ web.

Khi nhận được dữ liệu từ một máy chủ web, dữ liệu luôn luôn là một chuỗi.

Phân tích dữ liệu với JSON.parse(), các dữ liệu sẽ trở thành một đối tượng JavaScript.

* Ví dụ - phân tích JSON

Hãy tưởng tượng chúng ta nhận được văn bản này từ một máy chủ web :

'{ "name":"John", "age":30, "city":"New York"}'

Sử dụng hàm JavaScript JSON.parse() để chuyển văn bản thành một đối tượng JavaScript :

var obj = JSON.parse('{ "name":"John", "age":30, "city":"New York"}');

Chú ý : Đảm bảo rằng văn bản được viết bằng định dạng JSON, hoặc nếu không bạn sẽ nhận được một lỗi cú pháp.

Sử dụng các đối tượng JavaScript trong trang của bạn :

<p id="demo"></p>  
<script>  
document.getElementById("demo").innerHTML = obj.name + ", " + obj.age;  
</script>

* JSON từ máy chủ

Bạn có thể yêu cầu JSON từ máy chủ bằng cách sử dụng một yêu cầu AJAX.

Chừng nào phản hồi từ máy chủ được viết bằng định dạng JSON, bạn có thể phân tích các chuỗi thành một đối tượng JavaScript.

Ví dụ : sử dụng XMLHttpRequest để lấy dữ liệu từ máy chủ.

var xmlhttp = new XMLHttpRequest();  
xmlhttp.onreadystatechange = function() {  
  if (this.readyState == 4 && this.status == 200) {  
    var myObj = JSON.parse(this.responseText);  
    document.getElementById("demo").innerHTML = myObj.name;  
  }  
};  
xmlhttp.open("GET", "json\_demo.txt", true);  
xmlhttp.send();

* Mảng giống như JSON

Khi sử dụng JSON.parse() trên một JSON có nguồn gốc từ một mảng, phương pháp này sẽ trả về một mảng JavaScript thay vì một đối tượng JavaScript.

Ví dụ : JSON trở về từ máy chủ là một mảng.

var xmlhttp = new XMLHttpRequest();  
xmlhttp.onreadystatechange = function() {  
  if (this.readyState == 4 && this.status == 200) {  
    var myArr = JSON.parse(this.responseText);  
    document.getElementById("demo").innerHTML = myArr[0];  
  }  
};  
xmlhttp.open("GET", "json\_demo\_array.txt", true);  
xmlhttp.send();

* Trường hợp ngoại lệ

Phân tích Ngày

Đối tượng Date không được cho phép trong JSON.

Nếu bạn cần phải chứa một ngày, hãy viết nó như là một chuỗi.

Bạn có thể chuyển đổi nó rở thành một đối tượng ngày sau đó :

var text = '{ "name":"John", "birth":"1986-12-14", "city":"New York"}';  
var obj = JSON.parse(text);  
obj.birth = new Date(obj.birth);  
  
document.getElementById("demo").innerHTML = obj.name + ", " + obj.birth;

Hoặc bạn có thể sử dụng tham số thứ hai, các hàm JSON.parse(), được gọi là người làm sống lại.

Các người làm sống lại tham số là một chức năng để kiểm tra từng thuộc tính, trước khi trả về giá trị.

var text = '{ "name":"John", "birth":"1986-12-14", "city":"New York"}';  
var obj = JSON.parse(text, function (key, value) {  
  if (key == "birth") {  
    return new Date(value);  
  } else {  
    return value;  
  }  
});  
  
document.getElementById("demo").innerHTML = obj.name + ", " + obj.birth;

* Parsing Functions

Các hàm là không được phép trong JSON.

Nếu bạn cần phải chứa một hàm, hãy viết nó như là một chuỗi.

Bạn có thể chuyển đổi nó trở lại một hàm như sau :

var text = '{ "name":"John", "age":"function () {return 30;}", "city":"New York"}';  
var obj = JSON.parse(text);  
obj.age = eval("(" + obj.age + ")");  
document.getElementById("demo").innerHTML = obj.name + ", " + obj.age();

Bạn nên tránh sử dụng các hàm trong JSON, các hàm sẽ mất phạm vi của chúng, và bạn sẽ phải sử dụng eval() để chuyển đổi chúng trở lại vào hàm.

* Trình duyệt hỗ trợ

Hàm JSON.parse() được bao hàm tất cả các trình duyệt chính và trong các phiên bản mới nhất của ECMAScript (JavaScript).

Đối với các trình duyệt cũ, thư viện JavaScript có sẵn tại https://github.com/douglascrockford/JSON-js .

#### JSON Stringify

JSON thường được sử dụng để trao đổi dữ liệu đến/từ một máy chủ web.

Khi gửi dữ liệu đến một máy chủ web, dữ liệu phải là một chuỗi.

Chuyển đổi một đối tượng JavaScript thành một chuỗi với JSON.stringify().

* Stringify a JavaScript Object

Hãy tưởng tượng chúng ta có đối tượng này trong JavaScript :

var obj = { name: "John", age: 30, city: "New York" };

Sử dụng hàm JSON.stringify() để chuyển đổi nó thành một chuỗi :

var myJSON = JSON.stringify(obj);

Kết quả sẽ là một chuỗi. myJSON bây giờ là một chuỗi, và sẵn sàng để được gửi đến máy chủ.

var obj = { name: "John", age: 30, city: "New York" };  
var myJSON = JSON.stringify(obj);  
document.getElementById("demo").innerHTML = myJSON;

* Stringify a JavaScript Array

Hãy tưởng tượng chúng ta có một mảng trong JavaScript :

var arr = [ "John", "Peter", "Sally", "Jane" ];

Sử dụng hàm JSON.stringify() để chuyển nó thành một chuỗi :

var myJSON = JSON.stringify(arr);

myJSON bây giờ là một chuỗi, và sẵn sàng để được gửi đến máy chủ :

var arr = [ "John", "Peter", "Sally", "Jane" ];  
var myJSON = JSON.stringify(arr);  
document.getElementById("demo").innerHTML = myJSON;

* Exceptions
* Stringify Dates

Trong JSON, đối tượng date không hợp lệ. Các hàm JSON.stringify() sẽ chuyển đổi ngày thành một chuỗi.

var obj = { name: "John", today: new Date(), city : "New York" };  
var myJSON = JSON.stringify(obj);  
  
document.getElementById("demo").innerHTML = myJSON;

Bạn có thể chuyển đổi chuỗi thành một đối tượng date tại máy thu.

* Hàm stringify

Trong JSON, hàm không được cho phép như các giá trị đối tượng.

Các hàm JSON.stringify() sẽ loại bỏ bất kỳ hàm từ một đối tượng JavaScript, cả từ khóa và giá trị :

var obj = { name: "John", age: function () {return 30;}, city: "New York"};  
var myJSON = JSON.stringify(obj);  
document.getElementById("demo").innerHTML = myJSON;

Điều này có thể được bỏ qua nếu bạn chuyển đổi hàm của bạn thành chuỗi trước khi chạy hàm JSON.stringify().

var obj = { name: "John", age: function () {return 30;}, city: "New York" };  
obj.age = obj.age.toString();  
var myJSON = JSON.stringify(obj);  
  
document.getElementById("demo").innerHTML = myJSON;

Bạn nên tránh sử dụng các hàm trong JSON, các hàm sẽ mất phạm vi của chúng, và bạn sẽ phải sử dụng eval() để chuyển đổi chúng trở lại vào hàm.

* Trình duyệt hỗ trợ

Hàm JSON.stringify() được bao hàm tất cả các trình duyệt chính và trong các phiên bản mới nhất của ECMAScript (JavaScript).

#### JSON Objects

* Cú pháp

{ "name":"John", "age":30, "car":null }

Đối tượng JSON được bao quanh bởi dấu ngoặc nhọn {}

Đối tượng JSON được viết bằng từ khóa/giá trị

Khóa phải là kiểu chuỗi, và giá trị phải là một kiểu dữ liệu JSON hợp lệ (string, number, array, Boolean hoặc null).

Khóa và các giá trị được phân cách nhau bằng dấu hai chấm.

Mỗi cặp khóa/giá trị được phân cách bằng dấu phẩy.

* Truy cập vào các giá trị Object

Bạn có thể truy cập vào các giá trị của đối tượng bằng cách sử dụng ký hiệu dấu chấm (.) :

myObj = { "name":"John", "age":30, "car":null };  
x = myObj.name;

Bạn cũng có thể truy cập vào các giá trị của đối tượng bằng cách sử dụng ký hiệu khung ([]) :

myObj = { "name":"John", "age":30, "car":null };  
x = myObj["name"];

* Looping an Object

Bạn có thể lặp qua các thuộc tính của đối tượng bằng cách sử dụng voàng lặp for-in :

myObj = { "name":"John", "age":30, "car":null };  
for (x in myObj) {  
  document.getElementById("demo").innerHTML += x;  
}

Trong vòng lặp for-in, sử dụng ký hiệu khung để truy cập đến các giá trị của thuộc tính :

myObj = { "name":"John", "age":30, "car":null };  
for (x in myObj) {  
  document.getElementById("demo").innerHTML += myObj[x];  
}

* Nested JSON Object

Giá trị trong một đối tượng JSON có thể là một đối tượng JSON khác.

myObj = {  
  "name":"John",  
  "age":30,  
  "cars": {  
    "car1":"Ford",  
    "car2":"BMW",  
    "car3":"Fiat"  
  }  
 }

Bạn có thể truy cập vào các đối tượng JSON lồng nhau bằng cách sử dụng các ký hiệu dấu chấm hoặc ký hiệu khung :

x = myObj.cars.car2;  
// or:  
x = myObj.cars["car2"];

* Sửa đổi giá trị

Bạn có thể sử dụng ký hiệu dấu chấm để sửa đổi bất kỳ giá trị trong một đối tượng JSON :

myObj.cars.car2 = "Mercedes";

Bạn cũng có thể sử dụng ký hiệu khung để sửa đổi một giá trị trong một đối tượng JSON:

myObj.cars["car2"] = "Mercedes";

* Xóa thuộc tính của đối tượng

Sử dụng từ khóa delete để xóa các thuộc tính từ một đối tượng JSON :

delete myObj.cars.car2;

#### JSON Arrays

* Mảng giống như đối tượng JSON

[ "Ford", "BMW", "Fiat" ]

Mảng trong JSON gần giống như mảng trong JavaScript.

Trong JSON, giá trị mảng phải là kiểu string, number, object, array, Boolean hoặc null.

Trong JavaScript, giá trị mảng có thể là tất cả các kiểu đữ liệu trên, cùng với bất kỳ hành động JavaScript hợp lệ khác, bao gồm các hàm, date và không xác định.

* Mảng trong đối tượng JSON

Mảng có thể là giá trị của một thuộc tính của đối tượng:

{  
"name":"John",  
"age":30,  
"cars":[ "Ford", "BMW", "Fiat" ]  
}

* Truy cập vào giá trị của mảng

Bạn có thể truy cập vào các giá trị của mảng bằng cách sử dụng các chỉ số:

x = myObj.cars[0];

* Lặp qua một mảng

Bạn có thể truy cập vào các giá trị mảng bằng cách sự dụng vòng lặp for-in:

for (i in myObj.cars) {  
  x += myObj.cars[i];  
}

Hoặc bạn có thể sử dụng một vòng lặp for :

for (i = 0; i < myObj.cars.length; i++) {  
  x += myObj.cars[i];  
}

* Mảng lồng nhau trong đối tượng JSON

Giá trị trong một mảng cũng có thể là một mảng, hoặc thậm chí là một đối tượng JSON khác.

myObj = {  
  "name":"John",  
  "age":30,  
  "cars": [  
    { "name":"Ford", "models":[ "Fiesta", "Focus", "Mustang" ] },  
    { "name":"BMW", "models":[ "320", "X3", "X5" ] },  
    { "name":"Fiat", "models":[ "500", "Panda" ] }  
  ]  
 }

Để truy cập vào mảng trong mảng, sử dụng một vòng lặp for-in trong mỗi mảng :

for (i in myObj.cars) {  
  x += "<h1>" + myObj.cars[i].name + "</h1>";  
  for (j in myObj.cars[i].models) {  
    x += myObj.cars[i].models[j];  
  }  
}

* Thay đổi giá trị của mảng

Sử dụng chỉ số để sửa đổi một mảng :

myObj.cars[1] = "Mercedes";

* Xóa mục trong mảng

Sử dụng từ khóa delete để xóa các mục từ một mảng :

delete myObj.cars[1];

#### JSON PHP

* Tập tin PHP

PHP có một số hàm built-in để xử lý JSON.

Các đối tượng trong PHP có thể được chuyển đổi thành JSON bằng cách sử dụng hàm PHP json\_encode() :

<?php  
$myObj->name = "John";  
$myObj->age = 30;  
$myObj->city = "New York";  
$myJSON = json\_encode($myObj);  
echo $myJSON;  
?>

* Máy khách JavaScript

Đây là một JavaScript trên máy khách, sử dụng một cuộc gọi AJAX để yêu cầu các tập tin PHP từ ví dụ trên :

Sử dụng JSON.parse() để chuyển đổi kết quả vào một đối tượng Javascript:

var xmlhttp = new XMLHttpRequest();  
xmlhttp.onreadystatechange = function() {  
  if (this.readyState == 4 && this.status == 200) {  
    var myObj = JSON.parse(this.responseText);  
    document.getElementById("demo").innerHTML = myObj.name;  
  }  
};  
xmlhttp.open("GET", "demo\_file.php", true);  
xmlhttp.send();

* Mảng PHP

Mảng trong PHP cũng sẽ được chuyển đổi thành JSON khi sử dụng hàm PHP json\_encode() :

<?php  
$myArr = array("John", "Mary", "Peter", "Sally");  
$myJSON = json\_encode($myArr);  
echo $myJSON;  
?>

* Máy khách JavaScript

Đây là một đoạn mã JavaScript trên máy khách, sử dụng một hàm gọi AJAX để yêu cầu file PHP từ mảng trong ví dụ trên :

Sử dụng JSON.parse() để chuyển đổi kết quả vào một mảng JavaScript:

var xmlhttp = new XMLHttpRequest();  
xmlhttp.onreadystatechange = function() {  
  if (this.readyState == 4 && this.status == 200) {  
    var myObj = JSON.parse(this.responseText);  
    document.getElementById("demo").innerHTML = myObj[2];  
  }  
};  
xmlhttp.open("GET", "demo\_file\_array.php", true);  
xmlhttp.send();

* Cơ sở dữ liệu PHP

PHP là một ngôn ngữ lập trình phái máy chủ, nên được sử dụng cho các hoạt động mà chỉ có thể thwucj hiện bởi một máy chủ, như truy cập vào một cơ sở dữ liệu.

Hãy tưởng tượng bạn có một cơ sở dữ liệu trên máy chủ, chứa khách hàng, sản phẩm và nhà cung cấp.

Bạn muốn thực hiện một yêu cầu đến máy chủ để lấy 10 hồ sơ đầu tiên trong bảng “customer” :

Sử dụng JSON.stringify() để chuyển đối tượng JavaScript vào JSON:

obj = { "table":"customers", "limit":10 };  
dbParam = JSON.stringify(obj);  
xmlhttp = new XMLHttpRequest();  
xmlhttp.onreadystatechange = function() {  
  if (this.readyState == 4 && this.status == 200) {  
    document.getElementById("demo").innerHTML = this.responseText;  
  }  
};  
xmlhttp.open("GET", "json\_demo\_db.php?x=" + dbParam, true);  
xmlhttp.send();

Giải thích:

* Xác định một đối tượng chứa một bảng thuộc tính và một giới hạn các thuộc tính.
* Chuyển đổi đối tượng vào một chuỗi JSON.
* Gửi yêu cầu đến tập tin PHP, với chuỗi JSON như một tham số.
* Chờ đến khi yêu cầu được trả về với kết quả ( như JSON).
* Hiển thị các kết quả nhận được từ các file PHP.

Hãy xem các file PHP:

<?php  
header("Content-Type: application/json; charset=UTF-8");  
$obj = json\_decode($\_GET["x"], false);  
  
$conn = new mysqli("myServer", "myUser", "myPassword", "Northwind");  
$stmt = $conn->prepare("SELECT name FROM ? LIMIT ?");  
$stmt->bind\_param("ss", $obj->table, $obj->limit);  
$stmt->execute();  
$result = $stmt->get\_result();  
$outp = $result->fetch\_all(MYSQLI\_ASSOC);  
  
echo json\_encode($outp);  
?>

Giải thích tâp tin PHP

* Chuyển đổi theo yêu cầu vào một đối tượng, sử dụng hàm PHP json\_encode().
* Truy cập cơ sở dữ liệu, và điền vào một mảng với các dữ liệu được yêu cầu.
* Thêm mảng đến một đối tượng, và trả về đối tượng giống như JSON sử dụng hàm json\_encode().
* Lặp thông qua kết quả

Chuyển đổi két quả nhận được từu tập tin PHP vào đối tượng JavaScript, hoặc trong trường hợp này, vào một mảng JavaScript :

...  
xmlhttp.onreadystatechange = function() {  
  if (this.readyState == 4 && this.status == 200) {  
    myObj = JSON.parse(this.responseText);  
    for (x in myObj) {  
      txt += myObj[x].name + "<br>";  
    }  
    document.getElementById("demo").innerHTML = txt;  
  }  
};  
...

* PHP phương thức POST

Khi gửi dữ liệu đến máy chủ, cách tốt nhất là sử dụng phương pháp Http POST.

Để gửi yêu cầu AJAX bằng cách sử dụng phương thức POST, xác định phương pháp, và tiêu đề chính xác.

Các dữ liệu được gửi đến máy chủ bây giờ phải có đối số với phương thức send() :

obj = { "table":"customers", "limit":10 };  
dbParam = JSON.stringify(obj);  
xmlhttp = new XMLHttpRequest();  
xmlhttp.onreadystatechange = function() {  
  if (this.readyState == 4 && this.status == 200) {  
    myObj = JSON.parse(this.responseText);  
    for (x in myObj) {  
      txt += myObj[x].name + "<br>";  
    }  
    document.getElementById("demo").innerHTML = txt;  
  }  
};  
xmlhttp.**open**(**"POST"**, **"json\_demo\_db\_post.php"**, **true**);xmlhttp.**setRequestHeader**(**"Content-type"**, **"application/x-www-form-urlencoded"**);xmlhttp.**send**(**"x="** + dbParam);

Điều khác biệt duy nhất trong file PHP là phương thức để nhận dữ liệu được chuyển.

File PHP sử dụng $\_POST thay vì $\_GET :

<?php  
header("Content-Type: application/json; charset=UTF-8");  
$obj = json\_decode(**$\_POST**[**"x"**], false);  
$conn = new mysqli("myServer", "myUser", "myPassword", "Northwind");  
$stmt = $conn->prepare("SELECT name FROM ? LIMIT ?");  
$stmt->bind\_param("ss", $obj->table, $obj->limit);  
$stmt->execute();  
$result = $stmt->get\_result();  
$outp = $result->fetch\_all(MYSQLI\_ASSOC);  
  
echo json\_encode($outp);  
?>

#### JSON HTML

JSON có thể dễ dàng được chuyển qua JavaScript.

JavaScript có thể được sử dụng để tạo HTML trong trang web của bạn.

* HTML Table

Tạo một bảng HTML với dữ liệu nhận được dạng JSON:

obj = { table: "customers", limit: 20 };  
dbParam = JSON.stringify(obj);  
xmlhttp = new XMLHttpRequest();  
xmlhttp.onreadystatechange = function() {  
  if (this.readyState == 4 && this.status == 200) {  
    myObj = JSON.parse(this.responseText);  
    txt += "<table border='1'>"  
    for (x in myObj) {  
      txt += "<tr><td>" + myObj[x].name + "</td></tr>";  
    }  
    txt += "</table>"  
    document.getElementById("demo").innerHTML = txt;  
  }  
}  
xmlhttp.open("POST", "json\_demo\_db\_post.php", true);  
xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");  
xmlhttp.send("x=" + dbParam);

* Dynamic HTML Table

Tận dụng bảng HTML dựa trên giá trị của một danh sách thả xuống :



<select id="myselect" onchange="change\_myselect(this.value)">  
  <option value="">Choose an option:</option>  
  <option value="customers">Customers</option>  
  <option value="products">Products</option>  
  <option value="suppliers">Suppliers</option>  
</select>  
<script>  
function change\_myselect(sel) {  
  var obj, dbParam, xmlhttp, myObj, x, txt = "";  
  obj = { table: sel, limit: 20 };  
  dbParam = JSON.stringify(obj);  
  xmlhttp = new XMLHttpRequest();  
  xmlhttp.onreadystatechange = function() {  
    if (this.readyState == 4 && this.status == 200) {  
      myObj = JSON.parse(this.responseText);  
      txt += "<table border='1'>"  
      for (x in myObj) {  
        txt += "<tr><td>" + myObj[x].name + "</td></tr>";  
      }  
      txt += "</table>"  
      document.getElementById("demo").innerHTML = txt;  
    }  
  };  
  xmlhttp.open("POST", "json\_demo\_db\_post.php", true);  
  xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");  
  xmlhttp.send("x=" + dbParam);  
}  
</script>

* HTML Drop Down List

Tạo một danh sách thả xuống với dữ liệu nhận được là JSON:

obj = { table: "customers", limit: 20 };  
dbParam = JSON.stringify(obj);  
xmlhttp = new XMLHttpRequest();  
xmlhttp.onreadystatechange = function() {  
  if (this.readyState == 4 && this.status == 200) {  
    myObj = JSON.parse(this.responseText);  
    txt += "<select>"  
    for (x in myObj) {  
      txt += "<option>" + myObj[x].name;  
    }  
    txt += "</select>"  
    document.getElementById("demo").innerHTML = txt;  
  }  
}  
xmlhttp.open("POST", "json\_demo\_db\_post.php", true);  
xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");  
xmlhttp.send("x=" + dbParam);

#### JSON JSONP

JSONP là một phương thức để gửi dữ liệu JSON mà kho lo lắng về các vấn đề cross-doamin.

JSONP không sử dụng các đối tượng XMLHttpRequest.

JSONP sử dụng thẻ <script> để thay thế.

* Giới thiệu

JSONP là viết tắt của JSON với Padding.

Yêu cầu một file từ một miền khác có thể gây ra vấn đề, do chính sách cross-domain.

JSONP sử dụng lợi thế này, và yêu cầu các tập tin bằng cách sử dụng thẻ script thay cho đối tượng XMLHttpRequest.

<script src="demo\_jsonp.php">

* Server file

Tập tin trên máy chủ bao bọc kết quả bên trong một hàm gọi :

<?php  
$myJSON = '{ "name":"John", "age":30, "city":"New York" }';  
echo "myFunc(".$myJSON.");";  
?>

Kết quả trả về một cuộc gọi đến hàm có tên “myFunc” với dữ liệu JSON như một tham số.

Hãy chắc chắn rằng hàm tồn tại trên máy khách.

* Các hàm JavaScript

Hàm có tên là “myFunc” được đặt trên máy khách, và sẵn sàng xử lý dữ liệu JSON.

function myFunc(myObj) {  
  document.getElementById("demo").innerHTML = myObj.name;  
}

* Tạo một thẻ Script động

Ví dụ trên sẽ thực hiện hàm “myFunc” khi trang đang tải, dựa vào nơi bạn đặt thẻ script, điều này không thỏa mãn lắm.

Thẻ script chỉ nên được tạo khi cần thiết.

function clickButton() {  
  var s = document.createElement("script");  
  s.src = "demo\_jsonp.php";  
  document.body.appendChild(s);  
}

* Kết quả JSONP động

Các ví dụ trên vẫn còn rất tĩnh.

Hãy thử một ví dụ động bằng cách gửi JSON đến tập tin PHP, và để tập tin PHP trả về một đối tượng JSON dựa trên các thông tin nó nhận được.

PHP file :

<?php  
header("Content-Type: application/json; charset=UTF-8");  
$obj = json\_decode($\_GET["x"], false);  
$conn = new mysqli("myServer", "myUser", "myPassword", "Northwind");  
$result = $conn->query("SELECT name FROM ".$obj->$table." LIMIT ".$obj->$limit);  
$outp = array();  
$outp = $result->fetch\_all(MYSQLI\_ASSOC);  
  
echo "myFunc(".json\_encode($outp).")";  
?>

Giải thích :

* Chuyển đổi yêu cầu vào một đối tượng, sử dụng hàm PHP json\_decode().
* Truy cập cơ sở dữ liệu, và điền vào một mảng với các dữ liệu được yêu cầu.
* Thêm mảng cho một đối tượng.
* Chuyển đổi mảng vào JSON sử dụng hàm json\_encode().
* Bao bọc “myFunc” xung quanh đối tượng được trả về.
* Ví dụ JavaScript

Hàm “myFunc” sẽ được gọi từ tập tin PHP:

function clickButton() {  
  var obj, s  
  obj = { table: "products", limit: 10 };  
  s = document.createElement("script");  
  s.src = "jsonp\_demo\_db.php?x=" + JSON.stringify(obj);  
  document.body.appendChild(s);  
}  
function myFunc(myObj) {  
  var x, txt = "";  
  for (x in myObj) {  
    txt += myObj[x].name + "<br>";  
  }  
  document.getElementById("demo").innerHTML = txt;  
}

* Hàm Callback

Khi bạn không có quyền kiểm soát các tập tin máy chủ, làm thế nào để bạn có được những tập tin máy chủ để gọi hàm?

Đôi khi các tập tin máy chủ cung cấp một hàm callback như một tham số:

function clickButton() {  
  var s = document.createElement("script");  
  s.src = "jsonp\_demo\_db.php?callback=myDisplayFunction";  
  document.body.appendChild(s);  
}

## 2.3 CSS

### CSS Tutorial(CSS)

CSS là ngôn ngữ mô tả phong cách của tài liệu HTML.

CSS mô tả cách các phần tử HTML sẽ được hiển thị.

Hướng dẫn này sẽ dạy bạn CSS từ cơ bản đến nâng cao.

1. Ví dụ

Hướng dẫn CSS này chứa hàng trăm ví dụ CSS.

Với trình chỉnh sửa trực tuyến của chúng tôi, bạn có thể chỉnh sửa CSS và nhấp vào nút để xem kết quả.

body {  
  background-color: lightblue;  
}  
  
h1 {  
  color: white;  
  text-align: center;  
}  
  
p {  
  font-family: verdana;  
  font-size: 20px;  
}

**2.3.1.1. CSS Introduction**

1. CSS là gì?

- CSS là viết tắt của Cascading Style Sheets

- CSS mô tả cách các phần tử HTML được hiển thị trên màn hình, giấy hoặc trong phương tiện khác

- CSS tiết kiệm rất nhiều công việc. Nó có thể kiểm soát bố cục của nhiều trang web cùng một lúc

- Các bảng định kiểu bên ngoài được lưu trữ trong các tệp CSS

2. CSS Demo - Một trang HTML - Nhiều kiểu!

Nếu bạn không biết HTML là gì, chúng tôi khuyên bạn nên đọc Hướng dẫn HTML của chúng tôi.

3. CSS tiết kiệm rất nhiều công việc!

Các định nghĩa kiểu thường được lưu trong các tệp .css bên ngoài.

Với tệp biểu định kiểu bên ngoài, bạn có thể thay đổi giao diện của toàn bộ trang web bằng cách thay đổi chỉ một tệp!

**2.3.1.2 Cú pháp CSS**

1. Cú pháp CSS

Bộ quy tắc CSS bao gồm bộ chọn và khối khai báo:

Bộ chọn trỏ đến phần tử HTML mà bạn muốn tạo kiểu.

Khối khai báo chứa một hoặc nhiều khai báo được phân tách bằng dấu chấm phẩy.

Mỗi khai báo bao gồm tên thuộc tính CSS và một giá trị, được phân tách bằng dấu hai chấm.

Một khai báo CSS luôn kết thúc bằng dấu chấm phẩy và các khối khai báo được bao quanh bởi các dấu ngoặc nhọn.

Ví dụ:

p {  
  color: red;  
  text-align: center;  
}

2. CSS Comments

Nhận xét được sử dụng để giải thích mã và có thể giúp ích khi bạn chỉnh sửa mã nguồn vào một ngày sau đó.

Bình luận bị bỏ qua bởi các trình duyệt.

Thí dụ

Nhận xét CSS bắt đầu bằng / \* và kết thúc bằng \* /. Nhận xét cũng có thể trải dài trên nhiều dòng:

p {  
  color: red;  
  /\* This is a single-line comment \*/  
  text-align: center;  
}  
  
/\* This is  
a multi-line  
comment \*/

**2.3.1.3. Bộ chọn CSS**

1. Bộ chọn CSS

Bộ chọn CSS được sử dụng để "tìm" (hoặc chọn) các thành phần HTML mà bạn muốn tạo kiểu.

Chúng tôi có thể chia các bộ chọn CSS thành năm loại:

Bộ chọn đơn giản (chọn thành phần dựa trên tên, id, lớp)

Bộ chọn kết hợp (chọn các yếu tố dựa trên mối quan hệ cụ thể giữa chúng)

Bộ chọn lớp giả (chọn các thành phần dựa trên một trạng thái nhất định)

Bộ chọn phần tử giả (chọn và định kiểu một phần của phần tử)

Bộ chọn thuộc tính (chọn thành phần dựa trên giá trị thuộc tính hoặc thuộc tính)

Trang này sẽ giải thích các bộ chọn đơn giản CSS.

Bộ chọn phần tử CSS

Bộ chọn phần tử chọn các phần tử HTML dựa trên tên phần tử.

Thí dụ

Tại đây, tất cả các phần tử <p> trên trang sẽ được căn giữa, với màu văn bản màu đỏ:

p {  
  text-align: center;  
  color: red;  
}

3. Bộ chọn id CSS

Bộ chọn id sử dụng thuộc tính id của một phần tử HTML để chọn một phần tử cụ thể.

Id của một thành phần là duy nhất trong một trang, vì vậy bộ chọn id được sử dụng để chọn một thành phần duy nhất!

Để chọn một phần tử có id cụ thể, hãy viết ký tự băm (#), theo sau là id của phần tử.

Thí dụ

Quy tắc CSS dưới đây sẽ được áp dụng cho phần tử HTML với id = "para1":

#para1 {  
  text-align: center;  
  color: red;  
}

Lưu ý: Tên id không thể bắt đầu bằng số!

4. Bộ chọn lớp CSS

Bộ chọn lớp chọn các thành phần HTML với thuộc tính lớp cụ thể.

Để chọn các phần tử với một lớp cụ thể, hãy viết một ký tự dấu chấm (.), Theo sau là tên lớp.

Thí dụ

Trong ví dụ này, tất cả các phần tử HTML có class = "centre" sẽ có màu đỏ và căn giữa:

.center {  
  text-align: center;  
  color: red;  
}

Bạn cũng có thể chỉ định rằng chỉ các phần tử HTML cụ thể mới bị ảnh hưởng bởi một lớp.

Thí dụ

Trong ví dụ này, chỉ các phần tử <p> với class = "centre" sẽ được căn giữa:

p.center {  
  text-align: center;  
  color: red;  
}

Các phần tử HTML cũng có thể tham chiếu đến nhiều hơn một lớp.

Thí dụ

Trong ví dụ này, phần tử <p> sẽ được tạo kiểu theo class = "centre" và to class = "Large":

<p class="center large">This paragraph refers to two classes.</p>

Lưu ý: Tên lớp không thể bắt đầu bằng số!

5. Bộ chọn phổ quát CSS

Bộ chọn phổ quát (\*) chọn tất cả các thành phần HTML trên trang.

Thí dụ

Quy tắc CSS dưới đây sẽ ảnh hưởng đến mọi thành phần HTML trên trang:

\* {  
  text-align: center;  
  color: blue;  
}

6. Bộ chọn nhóm CSS

Bộ chọn nhóm chọn tất cả các thành phần HTML có cùng định nghĩa kiểu.

Nhìn vào mã CSS sau (các phần tử h1, h2 và p có cùng định nghĩa kiểu):

Example:

h1 {  
  text-align: center;  
  color: red;  
}  
h2 {  
  text-align: center;  
  color: red;  
}  
p {  
  text-align: center;  
  color: red;  
}

Sẽ tốt hơn để nhóm các bộ chọn, để giảm thiểu mã.

Để chọn nhóm, tách từng bộ chọn bằng dấu phẩy.

Thí dụ

Trong ví dụ này, chúng tôi đã nhóm các bộ chọn từ mã ở trên:

h1, h2, p {  
  text-align: center;  
  color: red;  
}

**2.3.1.4 Cách thêm CSS**

Khi trình duyệt đọc biểu định kiểu, nó sẽ định dạng tài liệu HTML theo thông tin trong biểu định kiểu.

Ba cách để chèn CSS

Có ba cách để chèn một biểu định kiểu:

-CSS bên ngoài

-CSS nội bộ

-CSS nội tuyến

Với biểu định kiểu bên ngoài, bạn có thể thay đổi giao diện của toàn bộ trang web bằng cách thay đổi chỉ một tệp!

Mỗi trang HTML phải bao gồm một tham chiếu đến tệp biểu định kiểu bên ngoài bên trong phần tử <link>, bên trong phần đầu.

Thí dụ

Kiểu bên ngoài được xác định trong phần tử <link>, bên trong phần <head> của trang HTML:

<!DOCTYPE html>  
<html>  
<head>  
<link rel="stylesheet" type="text/css" href="mystyle.css">  
</head>  
<body>  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>  
</body>  
</html>

Một biểu định kiểu ngoài có thể được viết trong bất kỳ trình soạn thảo văn bản nào và phải được lưu bằng phần mở rộng .css.

Tệp .css bên ngoài không được chứa bất kỳ thẻ HTML nào.

Đây là cách tập tin "mystyle.css" trông như thế nào:

body {  
  background-color: lightblue;  
}  
h1 {  
  color: navy;  
  margin-left: 20px;  
}

Lưu ý: Không thêm khoảng trắng giữa giá trị thuộc tính và đơn vị (chẳng hạn như lề trái: 20 px;). Cách chính xác là: lề-trái: 20px;

* CSS nội bộ

Một biểu định kiểu nội bộ có thể được sử dụng nếu một trang HTML duy nhất có một kiểu duy nhất.

Kiểu bên trong được xác định bên trong phần tử <style>, bên trong phần đầu.

Thí dụ

Kiểu bên trong được xác định trong phần tử <style>, bên trong phần <head> của trang HTML:

<!DOCTYPE html>  
<html>  
<head>  
<style>  
body {  
  background-color: linen;  
}  
h1 {  
  color: maroon;  
  margin-left: 40px;  
}  
</style>  
</head>  
<body>  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>  
</body>  
</html>

* CSS nội tuyến

Một kiểu nội tuyến có thể được sử dụng để áp dụng một kiểu duy nhất cho một phần tử.

Để sử dụng kiểu nội tuyến, hãy thêm thuộc tính style cho thành phần có liên quan. Thuộc tính style có thể chứa bất kỳ thuộc tính CSS nào.

Thí dụ

Kiểu nội tuyến được xác định trong thuộc tính "kiểu" của phần tử có liên quan:

<!DOCTYPE html>  
<html>  
<body>  
<h1 style="color:blue;text-align:center;">This is a heading</h1>  
<p style="color:red;">This is a paragraph.</p>  
</body>  
</html>

Mẹo: Kiểu nội tuyến sẽ mất nhiều ưu điểm của biểu định kiểu (bằng cách trộn nội dung với bản trình bày). Sử dụng phương pháp này một cách tiết kiệm.

* Multiple Style Sheets

Nếu một số thuộc tính đã được xác định cho cùng một bộ chọn (phần tử) trong các biểu định kiểu khác nhau, giá trị từ biểu định kiểu đọc cuối cùng sẽ được sử dụng.

Giả sử rằng một biểu định kiểu ngoài có kiểu sau cho phần tử <h1>:

h1 {  
  color: navy;  
}

Sau đó, giả sử rằng một biểu định kiểu nội bộ cũng có kiểu sau cho phần tử <h1>:

h1 {  
  color: orange;     
}

Thí dụ

Nếu kiểu bên trong được xác định sau liên kết đến biểu định kiểu ngoài, các phần tử <h1> sẽ là "màu cam":

<head>  
<link rel="stylesheet" type="text/css" href="mystyle.css">  
<style>  
h1 {  
  color: orange;  
}  
</style>  
</head>

Thí dụ

Tuy nhiên, nếu kiểu bên trong được xác định trước liên kết đến biểu định kiểu ngoài, các phần tử <h1> sẽ là "hải quân":

<head>  
<style>  
h1 {  
  color: orange;  
}  
</style>  
<link rel="stylesheet" type="text/css" href="mystyle.css">  
</head>

* Thứ tự xếp tầng

Kiểu nào sẽ được sử dụng khi có nhiều hơn một kiểu được chỉ định cho một phần tử HTML?

Tất cả các kiểu trong một trang sẽ "xếp tầng" thành một biểu định kiểu "ảo" mới theo các quy tắc sau, trong đó số một có mức ưu tiên cao nhất:

Kiểu nội tuyến (bên trong một phần tử HTML)

Biểu định kiểu bên ngoài và bên trong (trong phần đầu)

Trình duyệt mặc định

Vì vậy, một kiểu nội tuyến có mức ưu tiên cao nhất và sẽ ghi đè các kiểu bên ngoài và bên trong và mặc định của trình duyệt.

**2.3.1.5 MÀU CSS**

Màu sắc được chỉ định bằng cách sử dụng tên màu được xác định trước hoặc các giá trị RGB, HEX, HSL, RGBA, HSLA.

1. Tên màu CSS

Trong CSS, một màu có thể được chỉ định bằng cách sử dụng tên màu

Giống trong HTML

2. Màu nền css

Bạn có thể đặt màu nền cho các thành phần HTML:

<h1 style="background-color:DodgerBlue;">Hello World</h1>  
<p style="background-color:Tomato;">Lorem ipsum...</p>

3. Màu văn bản

Ta có thể đặt màu cho văn bản

<h1 style="color:Tomato;">Hello World</h1>  
<p style="color:DodgerBlue;">Lorem ipsum...</p>  
<p style="color:MediumSeaGreen;">Ut wisi enim...</p>

4. Màu viền

Có thể đặt màu của đường viền.

<h1 style="border:2px solid Tomato;">Hello World</h1>  
<h1 style="border:2px solid DodgerBlue;">Hello World</h1>  
<h1 style="border:2px solid Violet;">Hello World</h1>

5. Giá trị màu của CSS

Trong CSS, màu sắc cũng có thể được chỉ định bằng các giá trị RGB, giá trị HEX, giá trị HSL, giá trị RGBA và giá trị HSLA:

<h1 style="background-color:rgb(255, 99, 71);">...</h1>  
<h1 style="background-color:#ff6347;">...</h1>  
<h1 style="background-color:hsl(9, 100%, 64%);">...</h1>  
<h1 style="background-color:rgba(255, 99, 71, 0.5);">...</h1>  
<h1 style="background-color:hsla(9, 100%, 64%, 0.5);">...</h1>

6. Giá trị CSS RGB

Trong CSS, một màu có thể được chỉ định làm giá trị RGB, sử dụng công thức này:

rgb ( đỏ, xanh lá cây , xanh dương )

Mỗi tham số (đỏ, xanh lá cây và xanh dương) xác định cường độ của màu trong khoảng từ 0 đến 255.

Ví dụ: rgb (255, 0, 0) được hiển thị màu đỏ, vì màu đỏ được đặt thành giá trị cao nhất (255) và các giá trị khác được đặt thành 0.

Để hiển thị màu đen, tất cả các tham số màu phải được đặt thành 0, như thế này: rgb (0, 0, 0).

Để hiển thị màu trắng, tất cả các tham số màu phải được đặt thành 255, như thế này: rgb (255, 255, 255).

7. Giá trị CSS HEX

Trong CSS, một màu có thể được chỉ định bằng cách sử dụng giá trị thập lục phân trong biểu mẫu:

# rrggbb

Trong đó rr (đỏ), gg (xanh lá cây) và bb (xanh lam) là các giá trị thập lục phân trong khoảng từ 00 đến ff (giống như số thập phân 0-255).

Ví dụ: # ff0000 được hiển thị màu đỏ, vì màu đỏ được đặt thành giá trị cao nhất (ff) và các giá trị khác được đặt thành giá trị thấp nhất (00).

8. Giá trị HSL

Trong CSS, một màu có thể được chỉ định bằng cách sử dụng màu sắc, độ bão hòa và độ sáng (HSL) ở dạng:

hsl ( màu sắc , độ bão hòa , độ sáng )

Hue là một độ trên bánh xe màu từ 0 đến 360. 0 là màu đỏ, 120 là màu xanh lá cây và 240 là màu xanh lam.

Độ bão hòa là một giá trị phần trăm, 0% có nghĩa là màu xám và 100% là màu đầy đủ.

Độ sáng cũng là một tỷ lệ phần trăm, 0% là màu đen, 50% không phải là sáng hay tối, 100% là màu trắng

- Độ bão hòa

Độ bão hòa có thể được mô tả là cường độ của một màu.

100% là màu thuần khiết, không có sắc độ xám

50% là 50% màu xám, nhưng bạn vẫn có thể nhìn thấy màu.

0% là màu xám hoàn toàn, bạn không thể nhìn thấy màu nữa.

- Độ nhẹ nhàng

Độ sáng của màu sắc có thể được mô tả là bạn muốn cung cấp bao nhiêu ánh sáng, trong đó 0% có nghĩa là không có ánh sáng (đen), 50% có nghĩa là 50% ánh sáng (không tối cũng không sáng) 100% có nghĩa là ánh sáng đầy đủ (trắng).

Giá trị RGBA

Giá trị màu RGBA là phần mở rộng của giá trị màu RGB với kênh alpha - chỉ định độ mờ cho màu.

Giá trị màu RGBA được chỉ định bằng:

rgba ( đỏ, xanh lá cây , xanh dương, alpha )

Tham số alpha là một số trong khoảng 0,0 (hoàn toàn trong suốt) và 1.0 (hoàn toàn không minh bạch)

**2.3.1.6 NỀN CSS**

*Màu nền của CSS*

Các background-colortài sản quy định màu nền của một phần tử.

Với CSS, một màu thường được chỉ định bởi:

một tên màu hợp lệ - như "đỏ"

một giá trị HEX - như "# ff0000"

một giá trị RGB - như "rgb (255,0,0)"

Nhìn vào Giá trị màu CSS để biết danh sách đầy đủ các giá trị màu có thể.

Ví dụ:

body {  
  background-color: lightblue;  
}

Tại đây, các phần tử <h1>, <p> và <div> sẽ có các màu nền khác nhau:

h1 {  
  background-color: green;  
}  
div {  
  background-color: lightblue;  
}  
p {  
  background-color: yellow;  
}

*Hình nền CSS*

Các background-imagetài sản quy định cụ thể một hình ảnh để sử dụng như là nền tảng của một phần tử.

Theo mặc định, hình ảnh được lặp lại để nó bao phủ toàn bộ thành phần.

Ví dụ:

body {  
  background-image: url("paper.gif");  
}

*CSS lặp lại nền*

Theo mặc định, thuộc background-imagetính lặp lại một hình ảnh theo cả chiều ngang và chiều dọc.

Một số hình ảnh chỉ nên được lặp lại theo chiều ngang hoặc chiều dọc, nếu không chúng sẽ trông lạ, như thế này:

Ví dụ:

body {  
  background-image: url("gradient\_bg.png");  
}

*Lặp lại nền CSS: không lặp lại*

Chỉ hiển thị hình ảnh nền một lần cũng được chỉ định bởi thuộc background-repeattính:

Ví dụ:

body {  
  background-image: url("img\_tree.png");  
  background-repeat: no-repeat;  
}

*Vị trí nền CSS*

Các background-positiontài sản được sử dụng để xác định vị trí của các hình nền.

Ví dụ: Định vị hình ảnh nền ở góc trên bên phải:

body {  
  background-image: url("img\_tree.png");  
  background-repeat: no-repeat;  
  background-position: right top;  
}

*Phần đính kèm nền CSS*

Các background-attachmentbất động sản cho phéo hình nền nên di chuyển hoặc được cố định (sẽ không di chuyển với phần còn lại của trang)

Ví dụ: Chỉ định rằng hình nền phải được sửa

body {  
  background-image: url("img\_tree.png");  
  background-repeat: no-repeat;  
  background-position: right top;  
  background-attachment: fixed;  
}

*Nền CSS - Thuộc tính tốc ký*

Để rút ngắn mã, cũng có thể chỉ định tất cả các thuộc tính nền trong một thuộc tính duy nhất. Đây được gọi là một tài sản tốc ký.

Các tài sản tốc ký cho nền là background.

Khi sử dụng thuộc tính tốc ký, thứ tự của các giá trị thuộc tính là:

- background-color

- background-image

- background-repeat

- background-attachment

- background-position

Không có vấn đề gì nếu một trong các giá trị thuộc tính bị thiếu, miễn là các giá trị khác theo thứ tự này.

Ví dụ:

body {  
  background: #ffffff url("img\_tree.png") no-repeat right top;

Tất cả các thuộc tính nền CSS

**2.3.1.7 BIÊN GIỚI CSS**

*Thuộc tính viền CSS*

Các borderthuộc tính CSS cho phép bạn chỉ định kiểu dáng, chiều rộng và màu sắc của đường viền của một phần tử.

*Kiểu viền CSS*

Các border-styletài sản quy định cụ thể những loại biên giới để hiển thị.

Các giá trị sau được cho phép:

- dotted - Xác định đường viền chấm

- dashed - Xác định đường viền nét đứt

- solid - Xác định đường viền chắc chắn

- double - Xác định đường viền đôi

- groove- Xác định đường viền có rãnh 3D. Hiệu ứng phụ thuộc vào giá trị màu viền

- ridge- Xác định đường viền có viền 3D. Hiệu ứng phụ thuộc vào giá trị màu viền

- inset- Xác định đường viền 3D. Hiệu ứng phụ thuộc vào giá trị màu viền

- outset- Xác định đường viền đầu 3D. Hiệu ứng phụ thuộc vào giá trị màu viền

- none - Xác định không có biên giới

- hidden - Xác định đường viền ẩn

Các border-stylebất động sản có thể có từ một đến bốn giá trị (đối với biên giới hàng đầu, biên phải, biên giới phía dưới, và biên giới trái).

Ví dụ:Trình diễn các kiểu viền khác nhau:

p.dotted {border-style: dotted;}  
p.dashed {border-style: dashed;}  
p.solid {border-style: solid;}  
p.double {border-style: double;}  
p.groove {border-style: groove;}  
p.ridge {border-style: ridge;}  
p.inset {border-style: inset;}  
p.outset {border-style: outset;}  
p.none {border-style: none;}  
p.hidden {border-style: hidden;}  
p.mix {border-style: dotted dashed solid double;}

Kết quả:

*Độ rộng đường viền CSS*

Tài border-widthsản chỉ định chiều rộng của bốn biên giới.

Chiều rộng có thể được đặt thành một kích thước cụ thể (tính bằng px, pt, cm, em, v.v.) hoặc bằng cách sử dụng một trong ba giá trị được xác định trước: mỏng, trung bình hoặc dày.

Các border-widthbất động sản có thể có từ một đến bốn giá trị (đối với biên giới hàng đầu, biên phải, biên giới phía dưới, và biên giới trái)

Ví dụ:

p.one {  
  border-style: solid;  
  border-width: 5px;  
}  
p.two {  
  border-style: solid;  
  border-width: medium;  
}  
p.three {  
  border-style: solid;  
  border-width: 2px 10px 4px 20px;  
}

*Màu viền CSS*

Các border-colortài sản được sử dụng để thiết lập màu sắc của bốn biên giới.

Màu sắc có thể được thiết lập bởi:

- Tên - chỉ định tên màu, như "đỏ"

- Hex - chỉ định giá trị hex, như "# ff0000"

- RGB - chỉ định giá trị RGB, như "rgb (255,0,0)"

trong suốt

Các border-colorbất động sản có thể có từ một đến bốn giá trị (đối với biên giới hàng đầu, biên phải, biên giới phía dưới, và biên giới trái).

Nếu border-colorkhông được đặt, nó sẽ kế thừa màu của phần tử.

Ví dụ:

p.one {  
  border-style: solid;  
  border-color: red;  
}  
p.two {  
  border-style: solid;  
  border-color: green;  
}  
p.three {  
  border-style: solid;  
  border-color: red green blue yellow;  
}

*CSS Border - Các mặt riêng lẻ*

Từ các ví dụ trên bạn đã thấy rằng có thể chỉ định một đường viền khác nhau cho mỗi bên.

Trong CSS, cũng có các thuộc tính để chỉ định từng đường viền (trên, phải, dưới và trái)

Vì vậy, đây là cách nó hoạt động:

Nếu thuộc border-styletính có bốn giá trị:

· kiểu viền: chấm đôi nét đứt;

- đường viền trên cùng được chấm

- đường viền bên phải là vững chắc

- viền dưới là gấp đôi

- đường viền bên trái bị đứt

Nếu thuộc border-styletính có ba giá trị:

· kiểu viền: chấm đôi rắn chắc;

- đường viền trên cùng được chấm

- biên giới bên phải và bên trái là vững chắc

- viền dưới là gấp đôi

Nếu thuộc border-styletính có hai giá trị:

· kiểu viền: chấm rắn;

- viền trên và dưới được chấm

- biên giới bên phải và bên trái là vững chắc

Nếu border-styletài sản có một giá trị:

· kiểu viền: chấm chấm;

- tất cả bốn biên giới được chấm

Các border-styletài sản được sử dụng trong ví dụ trên. Tuy nhiên, nó cũng hoạt động với border-width và border-color.

Ví dụ:

p {  
  border-top-style: dotted;  
  border-right-style: solid;  
  border-bottom-style: dotted;  
  border-left-style: solid;  
}

*CSS Border - Thuộc tính tốc ký*

Như bạn có thể thấy từ các ví dụ trên, có nhiều thuộc tính cần xem xét khi xử lý đường viền.

Để rút ngắn mã, cũng có thể chỉ định tất cả các thuộc tính viền riêng lẻ trong một thuộc tính.

Tài bordersản là một tài sản tốc ký cho các thuộc tính biên giới riêng lẻ sau đây:

- border-width

- border-style (cần thiết)

- border-color

ví dụ:

p {  
  border: 5px solid red;  
}

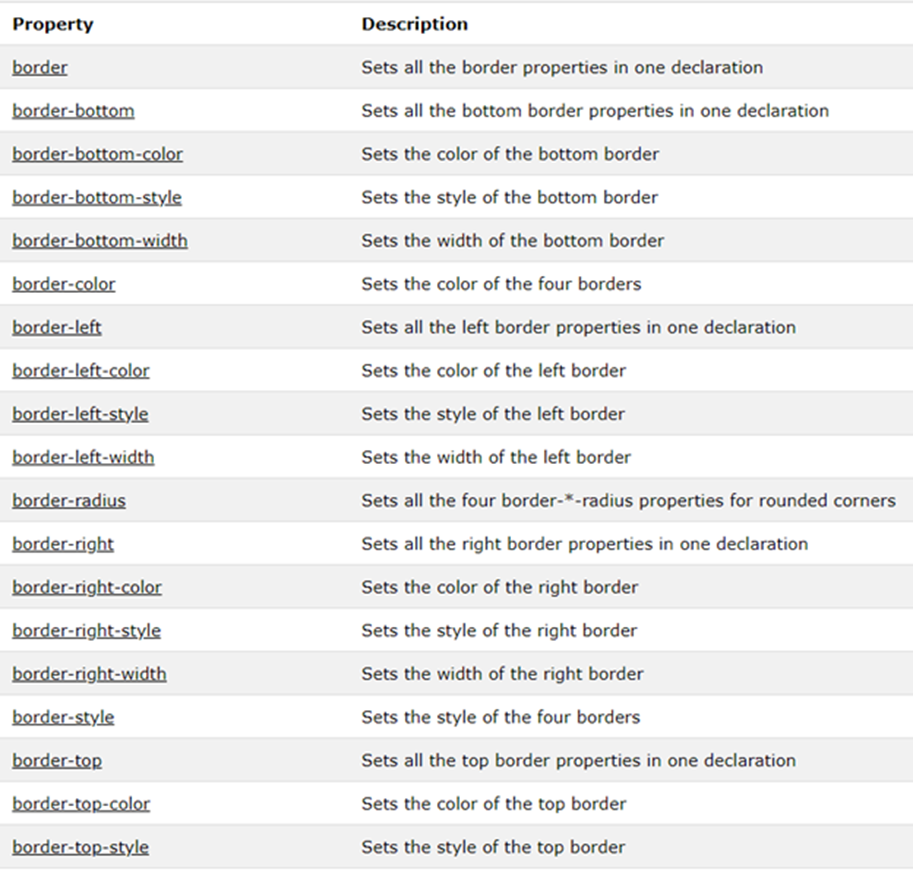
*Biên giới tròn CSS*

Các border-radiustài sản được sử dụng để thêm khung viền tròn đến một yếu tố

Ví dụ:

p {  
  border: 2px solid red;  
  border-radius: 5px;  
}

*Thuộc tính*

**

**2.3.1.8 Margin CSS**

Các marginthuộc tính CSS được sử dụng để tạo không gian xung quanh các phần tử, bên ngoài bất kỳ đường viền xác định nào.

Với CSS, bạn có toàn quyền kiểm soát lề. Có các thuộc tính để đặt lề cho mỗi bên của một phần tử (trên cùng, bên phải, dưới cùng và bên trái).

*Ký quỹ - Sides cá nhân*

CSS có các thuộc tính để chỉ định lề cho mỗi bên của một phần tử:

- margin-top

- margin-right

- margin-bottom

- margin-left

Tất cả các thuộc tính lề có thể có các giá trị sau:

- tự động - trình duyệt tính toán lề

- chiều dài - chỉ định lề trong px, pt, cm, v.v.

- % - chỉ định lề theo% chiều rộng của phần tử chứa

- kế thừa - chỉ định rằng lề phải được kế thừa từ phần tử cha

Ví dụ:

p {  
  margin-top: 100px;  
  margin-bottom: 100px;  
  margin-right: 150px;  
  margin-left: 80px;  
}

*Ký quỹ - Tài sản tốc ký*

Để rút ngắn mã, có thể chỉ định tất cả các thuộc tính lề trong một thuộc tính.

Tài marginsản là một thuộc tính tốc ký cho các thuộc tính ký quỹ riêng lẻ sau:

- margin-top

- margin-right

- margin-bottom

- margin-left

Vì vậy, đây là cách nó hoạt động:

Nếu thuộc margintính có bốn giá trị:

lề: 25px 50px 75px 100px;

- lề trên là 25px

- lề phải là 50px

- lề dưới là 75px

- lề trái là 100px

Nếu thuộc margintính có ba giá trị:

lề: 25px 50px 75px;

- lề trên là 25px

- lề phải và trái là 50px

- lề dưới là 75px

Nếu thuộc margintính có hai giá trị:

- lề: 25px 50px;

- lề trên và dưới là 25px

- lề phải và trái là 50px

Nếu margintài sản có một giá trị:

- lề: 25px;

tất cả bốn lề là 25px

*Giá trị tự động*

Bạn có thể đặt thuộc tính lề autothành trung tâm theo chiều ngang của phần tử trong vùng chứa của nó.

Phần tử sau đó sẽ chiếm chiều rộng được chỉ định và không gian còn lại sẽ được chia đều giữa lề trái và lề phải.

Ví dụ:

div {  
  width: 300px;  
  margin: auto;  
  border: 1px solid red;  
}

*Giá trị kế thừa*

Ví dụ này cho phép lề trái của phần tử <p class = "ex1"> được kế thừa từ phần tử cha (<div>):

Ví dụ: Sử dụng giá trị kế thừa

div {  
  border: 1px solid red;  
  margin-left: 100px;  
}  
p.ex1 {  
  margin-left: inherit;

*Ký quỹ giảm*

Lề trên và dưới của các yếu tố đôi khi được thu gọn thành một lề duy nhất bằng với lề lớn nhất trong hai lề.

Điều này không xảy ra ở lề trái và phải! Chỉ có lợi nhuận trên và dưới

Trong ví dụ trên, phần tử <h1> có lề dưới là 50px và phần tử <h2> có lề trên được đặt thành 20px.

Thông thường dường như sẽ gợi ý rằng lề dọc giữa <h1> và <h2> sẽ có tổng cộng 70px (50px + 20px). Nhưng do sự sụp đổ lề, lề thực tế kết thúc là 50px.

Tất cả các thuộc tính ký quỹ CSS

**2.3.1.9 ĐỆM CSS**

*Đệm CSS*

Các paddingthuộc tính CSS được sử dụng để tạo không gian xung quanh nội dung của một phần tử, bên trong bất kỳ đường viền được xác định nào.

Với CSS, bạn có toàn quyền kiểm soát phần đệm. Có các thuộc tính để thiết lập phần đệm cho mỗi bên của một phần tử (trên cùng, bên phải, dưới cùng và bên trái).

*Đệm - Các mặt riêng lẻ*

CSS có các thuộc tính để chỉ định phần đệm cho mỗi bên của một phần tử:

- padding-top

- padding-right

- padding-bottom

- padding-left

Tất cả các thuộc tính đệm có thể có các giá trị sau:

- chiều dài - chỉ định phần đệm trong px, pt, cm, v.v.

- % - chỉ định phần đệm theo% chiều rộng của phần tử chứa

- kế thừa - chỉ định rằng phần đệm nên được kế thừa từ phần tử cha

*Đệm - Tài sản tốc ký*

Để rút ngắn mã, có thể chỉ định tất cả các thuộc tính đệm trong một thuộc tính.

Tài paddingsản là một thuộc tính tốc ký cho các thuộc tính đệm riêng lẻ sau:

- padding-top

- padding-right

- padding-bottom

- padding-left

Vì vậy, đây là cách nó hoạt động:

Nếu thuộc paddingtính có bốn giá trị:

· phần đệm: 25px 50px 75px 100px;

- đệm hàng đầu là 25px

- đệm bên phải là 50px

- đệm dưới là 75px

- đệm bên trái là 100px

Nếu thuộc paddingtính có ba giá trị:

· phần đệm: 25px 50px 75px;

- đệm hàng đầu là 25px

- miếng đệm bên phải và bên trái là 50px

Nếu thuộc paddingtính có hai giá trị:

· phần đệm: 25px 50px;

- miếng đệm trên và dưới là 25px

- miếng đệm bên phải và bên trái là 50px- đệm dưới là 75px

Nếu paddingtài sản có một giá trị:

· phần đệm: 25px;

- tất cả bốn paddings là 25px

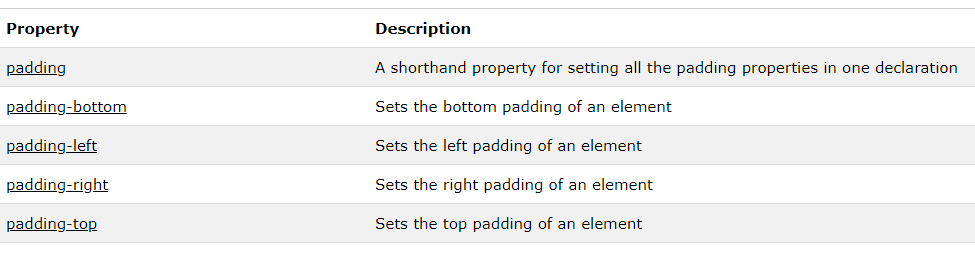
*Đệm và chiều rộng phần tử*

widthThuộc tính CSS chỉ định chiều rộng của vùng nội dung của phần tử. Vùng nội dung là phần bên trong phần đệm, đường viền và lề của một phần tử ( mô hình hộp ).

Vì vậy, nếu một phần tử có chiều rộng được chỉ định, phần đệm được thêm vào phần tử đó sẽ được thêm vào tổng chiều rộng của phần tử. Đây thường là một kết quả không mong muốn.

Để giữ chiều rộng ở 300px, bất kể số lượng đệm, bạn có thể sử dụng thuộc box-sizingtính. Điều này làm cho phần tử duy trì chiều rộng của nó; nếu bạn tăng phần đệm, không gian nội dung khả dụng sẽ giảm.

*Tất cả các thuộc tính đệm CSS*



***2.3.1.10 CHIỀU CAO VÀ CHIỀU RỘNG CSS***

*CSS Thiết lập chiều cao và chiều rộng*

Các thuộc tính heightvà widthđược sử dụng để đặt chiều cao và chiều rộng của một phần tử.

Các thuộc tính chiều cao và chiều rộng không bao gồm phần đệm, đường viền hoặc lề. Nó đặt chiều cao / chiều rộng của khu vực bên trong phần đệm, đường viền và lề của phần tử.

*Giá trị chiều cao / chiều rộng CSS*

Các thuộc tính heightvà widthcó thể có các giá trị sau:

- auto- Đây là mặc định. Trình duyệt tính toán chiều cao và chiều rộng

- length - Xác định chiều cao / chiều rộng tính bằng px, cm, v.v.

- % - Xác định chiều cao / chiều rộng tính bằng phần trăm của khối chứa

- initial - Đặt chiều cao / chiều rộng thành giá trị mặc định của nó

- inherit - Chiều cao / chiều rộng sẽ được kế thừa từ giá trị cha của nó

*Ví dụ chiều cao / chiều rộng CSS*

ví dụ

Đặt chiều cao và chiều rộng của phần tử <div>:

div {  
  height: 200px;  
  width: 50%;  
  background-color: powderblue;  
}

*Đặt chiều rộng tối đa*

Các max-widthtài sản được sử dụng để thiết lập chiều rộng tối đa của một phần tử.

Có max-widththể được chỉ định trong các giá trị độ dài , như px, cm, v.v. hoặc tính bằng phần trăm (%) của khối chứa hoặc được đặt thành không (đây là mặc định. Có nghĩa là không có chiều rộng tối đa).

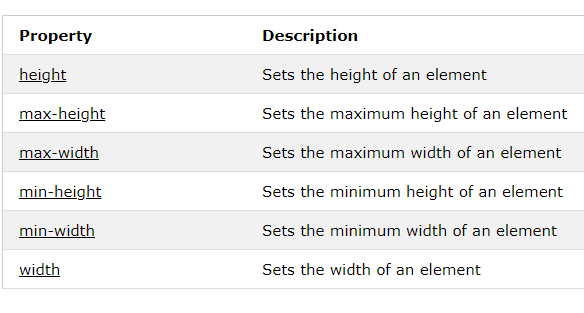
Sự cố <div>xảy ra ở trên xảy ra khi cửa sổ trình duyệt nhỏ hơn chiều rộng của phần tử (500px). Trình duyệt sau đó thêm một thanh cuộn ngang vào trang.

max-widthThay vào đó, sử dụng thay vào đó, sẽ cải thiện khả năng xử lý các cửa sổ nhỏ của trình duyệt.

ví dụ: Phần tử <div> này có chiều cao 100 pixel và chiều rộng tối đa 500 pixel

div {  
  max-width: 500px;  
  height: 100px;  
  background-color: powderblue;  
}

*Tất cả các thuộc tính kích thước CSS*



## 

### 2.3.1.11 MÔ HÌNH HỘP CSS

## Mô hình hộp CSS

## Tất cả các yếu tố HTML có thể được coi là hộp. Trong CSS, thuật ngữ "mô hình hộp" được sử dụng khi nói về thiết kế và bố cục.Mô hình hộp CSS về cơ bản là một hộp bao quanh mọi phần tử HTML. Nó bao gồm: lề, viền, phần đệm và nội dung thực tế. Hình ảnh dưới đây minh họa mô hình hộp:

## Giải thích về các phần khác nhau:

## · Nội dung - Nội dung của hộp, nơi xuất hiện văn bản và hình ảnh

## · Đệm - Xóa một khu vực xung quanh nội dung. Phần đệm trong suốt

## · Đường viền - Đường viền bao quanh phần đệm và nội dung

## · Ký quỹ - Xóa một khu vực bên ngoài biên giới. Lề là trong suốt

## Mô hình hộp cho phép chúng ta thêm một đường viền xung quanh các phần tử và để xác định khoảng trắng giữa các phần tử.

## Chiều rộng và chiều cao của một phần tử

## Để đặt chính xác chiều rộng và chiều cao của một phần tử trong tất cả các trình duyệt, bạn cần biết mô hình hộp hoạt động như thế nào.

## Ví dụ:

div {  
  width: 320px;  
  padding: 10px;  
  border: 5px solid gray;  
  margin: 0;  
}

### 2.3.1.12 PHÁC THẢO CSS

## Phác thảo CSS

## Một phác thảo là một đường được vẽ xung quanh các phần tử, BÊN NGOÀI các đường viền, để làm cho phần tử "nổi bật"

## CSS có các thuộc tính phác thảo sau:

## · outline-style

## · outline-color

## · outline-width

## · outline-offset

## · outline

## Phong cách phác thảo CSS

## Các outline-stylebất động sản xác định phong cách của đề cương, và có thể có một trong các giá trị sau:

## - dotted - Xác định một phác thảo chấm

## - dashed - Xác định một phác thảo nét

## - solid - Xác định một phác thảo vững chắc

## - double - Xác định một phác thảo kép

## - groove - Xác định đường viền có rãnh 3D

## - ridge - Xác định phác thảo có viền 3D

## - inset - Xác định phác thảo chèn 3D

## - - outset - Xác định phác thảo đầu ra 3D

## - none - Xác định không có phác thảo

## - hidden - Xác định một phác thảo ẩn

## Màu phác thảo CSS

## Các outline-colortài sản được sử dụng để thiết lập màu sắc của phác thảo.

## Màu sắc có thể được thiết lập bởi:

## - Tên - chỉ định tên màu, như "đỏ"

## - RGB - chỉ định giá trị RGB, như "rgb (255,0,0)"

## - Hex - chỉ định giá trị hex, như "# ff0000"

## - đảo ngược - thực hiện đảo ngược màu (đảm bảo có thể nhìn thấy đường viền, bất kể nền màu)

## Chiều rộng phác thảo CSS

## Các outline-widthtài sản quy định cụ thể rộng đường viền, và có thể có một trong các giá trị sau:

## - mỏng (thường là 1px)

## - trung bình (thường là 3px)

## - dày (thường là 5px)

## - Một kích thước cụ thể (tính bằng px, pt, cm, em, v.v.)

## 5CSS Outline - Thuộc tính tốc ký

## Các outlinebất động sản là một tài sản viết tắt cho thiết lập các thuộc tính outline cá nhân sau đây:

## - outline-width

## - outline-style (cần thiết)

## - outline-color

## Các outlinetài sản được quy định như một, hai, hoặc ba giá trị từ danh sách trên. Thứ tự của các giá trị không quan trọng.

## Bù đắp phác thảo CSS

## Các outline-offsetbất động sản cho biết thêm khoảng trống giữa một phác thảo và cạnh / biên giới của một phần tử. Không gian giữa một yếu tố và phác thảo của nó là trong suốt.

## Tất cả các thuộc tính phác thảo CSS

## 

**2.3.1.13 CSS TIÊU ĐỀ**

1. Văn bản màu

Các colortài sản được sử dụng để thiết lập màu sắc của văn bản. Màu sắc được chỉ định bởi:

- một tên màu - như "đỏ"

- một giá trị HEX - như "# ff0000"

- một giá trị RGB - như "rgb (255,0,0)"

Nhìn vào Giá trị màu CSS để biết danh sách đầy đủ các giá trị màu có thể.

Màu văn bản mặc định cho một trang được xác định trong bộ chọn cơ thể.

Ví dụ:

body {  
  color: blue;  
}  
h1 {  
  color: green;  
}

2. Căn chỉnh văn bản

Các text-aligntài sản được sử dụng để thiết lập sự liên kết ngang của một văn bản.

Một văn bản có thể được căn trái hoặc phải, căn giữa hoặc hợp lý.

Khi thuộc text-aligntính được đặt thành "biện minh", mỗi dòng được kéo dài sao cho mỗi dòng có chiều rộng bằng nhau và lề trái và phải thẳng (như trên tạp chí và báo)

Ví dụ:

h1 {  
  text-align: center;  
}  
h2 {  
  text-align: left;  
}  
h3 {  
  text-align: right;  
}

3. Trang trí văn bản

Tài text-decorationsản được sử dụng để thiết lập hoặc loại bỏ trang trí khỏi văn bản.

Giá trị text-decoration: none;thường được sử dụng để xóa gạch chân khỏi các liên kết

Các text-decorationgiá trị khác được sử dụng để trang trí văn bản:

Lưu ý: Không nên gạch chân văn bản không phải là một liên kết, vì điều này thường gây nhầm lẫn cho người đọc

4. Chuyển đổi văn bản

Các text-transformtài sản được sử dụng để xác định chữ hoa và chữ thường trong một văn bản.

Nó có thể được sử dụng để biến mọi thứ thành chữ in hoa hoặc chữ thường hoặc viết hoa chữ cái đầu tiên của mỗi từ

Ví dụ:

p.uppercase {  
  text-transform: uppercase;  
}  
p.lowercase {  
  text-transform: lowercase;  
}  
p.capitalize {  
  text-transform: capitalize;  
}

5.Văn bản thụt lề

Các text-indenttài sản được dùng để xác định thụt đầu dòng của dòng đầu tiên của một văn bản

VÍ DỤ:

p {  
  text-indent: 50px;  
}

6. Khoảng cách chữ

Các letter-spacingtài sản được sử dụng để xác định khoảng cách giữa các ký tự trong một văn bản.

Ví dụ sau minh họa cách tăng hoặc giảm khoảng trắng giữa các ký tự:

h1 {  
  letter-spacing: 3px;  
}  
  
h2 {  
  letter-spacing: -3px;  
}

7. Chiều cao giữa các dòng

Các line-heighttài sản được dùng để xác định khoảng cách giữa các dòng:

p.small {  
  line-height: 0.8;  
}  
p.big {  
  line-height: 1.8;  
}

8. Text directions

Các directiontài sản được sử dụng để thay đổi hướng văn bản của một phần tử

p {  
  direction: rtl;  
}

9.Khoảng cách từ

Các word-spacingtài sản được sử dụng để xác định khoảng cách giữa các từ trong một văn bản.

Ví dụ sau minh họa cách tăng hoặc giảm khoảng trắng giữa các từ:

h1 {  
  word-spacing: 10px;  
}  
h2 {  
  word-spacing: -5px;  
}

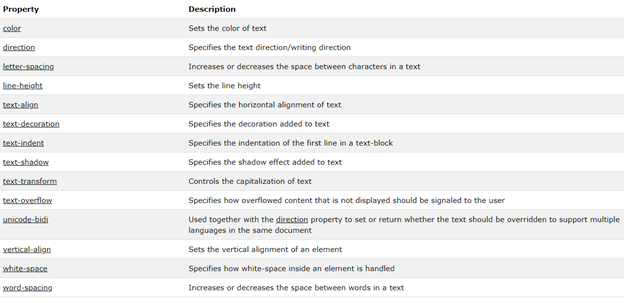
10.Bóng văn bản

Các text-shadowtài sản thêm bóng cho văn bản.

Ví dụ sau chỉ định vị trí của bóng ngang (3px), vị trí của bóng dọc (2px) và màu của bóng (đỏ):

h1 {  
  text-shadow: 3px 2px red;  
}

11.Tất cả các thuộc tính văn bản CSS

**

**2.3.1.14 CSS Fonts**

Để thiết lập font cho một trang HTML hoặc cho từng phần tử trong trang, bạn sử dụng các thuộc tính sau:

• Thuộc tính font-family: được sử dụng để thay đổi bề mặt font

• Thuộc tính font-style: được sử dụng để tạo một font chữ nghiêng hoặc chếch

• Thuộc tính font-variant: được sử dụng để tạo những chữ hoa nhỏ (small-cap)

• Thuộc tính font-weight: được sử dụng để tăng giảm độ đậm của font

• Thuộc tính font-size: được sử dụng để xác định kích cỡ font

• Thuộc tính font: sử dụng thuộc tính này nếu bạn muốn viết ít code hơn mà vẫn xác định được các thuộc tính liên quan tới font ở trên

*Thiết lập Font Family trong CSS*

Trong CSS, có hai loại Font Family:

• generic family: một nhóm các Font Family có bề mặt khá tương tự nhau. Ví dụ Serif hoặc Monospace.

• font family: một font family cụ thể, ví dụ như Time New Roman hoặc Arial.

Để thay đổi bề mặt font của một phần tử nào đó, bạn có thể sử dụng thuộc tính font-family trong CSS.

Nói chung, khi sử dụng thuộc tính font-family, bạn nên xác định nhiều hơn một giá trị. Nếu trình duyệt không hỗ trợ giá trị font đầu tiên, nó sẽ thử giá trị tiếp theo, … Bắt đầu với giá trị font mà bạn muốn và kết thúc với một generic family để trình duyệt chọn một font tương tự trong generic family nếu không có font nào có sẵn.

Ghi chú: Nếu tên font có nhiều hơn một từ, thì bạn phải đặt trong trích dẫn kép, ví dụ như: "Times New Roman".

Dưới đây là ví dụ minh họa cách xác định Font Family cho một phần tử. Giá trị cho thuộc tính font-family là bất kỳ tên font family hợp lệ nào.

1. p {
2. **font-family**: "Times New Roman", Times, serif;
3. }

*Font Style trong CSS*

Thuộc tính font-style trong CSS thường được sử dụng khi bạn muốn xác định một font chữ nghiêng. Các giá trị mà thuộc tính này có thể nhận là: normal, italic hoặc oblique. Trong đó, normal là hiển thị văn bản như bình thường, italic và oblique thì khá giống nhau, để hiển thị dạng nghiêng. Tuy nhiên, oblique ít được hỗ trợ hơn.

Ví dụ

1. p.normal {
2. **font-style**: normal;
3. }
5. p.italic {
6. **font-style**: italic;
7. }
9. p.oblique {
10. **font-style**: oblique;
11. }

*Thiết lập Font Variant trong CSS*

Để tạo các chữ hoa nhỏ (dạng small-cap), bạn sử dụng thuộc tính font-variant trong CSS. Thuộc tính này nhận hai giá trị là normal và small-caps. Ví dụ:

1. p.normal {
2. **font-variant**: normal;
3. }
5. p.small {
6. **font-variant**: small-caps;
7. }

*Thiết lập Font Weight trong CSS*

Để thiết lập mức độ đậm của văn bản, bạn sử dụng thuộc tính font-weight trong CSS. Thuộc tính này có thể nhận các giá trị: normal, bold, bolder, lighter, 100, 200, 300, 400, 500, 600, 700, 800, 900.

1. p.normal {
2. **font-weight**: normal;
3. }p.thick {
4. **font-weight**: bold;
5. }

*Thiết lập Font Size trong CSS*

Để thiết lập kích cỡ font, bạn sử dụng thuộc tính font-size trong CSS. Thuộc tính này có thể nhận các giá trị tương đối hoặc tuyệt đối: xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger, size in pixels hoặc in %.Ví dụ

1. **<html>**
2. **<head>**
3. **</head>**
4. **<body>**
5. **<p** style="font-size:20px;"**>**Doan van nay co Font Size la 20 pixel.**</p>**
6. **<p** style="font-size:small;"**>**Doan van nay co Font Size la small**</p>**
7. **<p** style="font-size:large;"**>**Doan van nay co Font Size la large**</p>**
8. **</body>**
9. **</html>**

Việc quản lý kích cỡ văn bản trong các phần khác nhau là rất quan trong trong thiết kế Web. Tuy nhiên, bạn không nên sử dụng thuộc tính font-size-adjust để làm cho các đoạn văn trông giống như các phần tiêu đề, hoặc tiêu đề lại trông giống như đoạn văn như trong phần Thuộc tính font-size-adjust trong CSS dưới đây. Lời khuyên hữu ích cho bạn là hãy sử dụng các thẻ HTML phù hợp, ví dụ như các thẻ <h1> - <h6> cho phần tiêu đề và thẻ <p> cho các đoạn văn.

*Thuộc tính font-size-adjust trong CSS*

Sử dụng thuộc tính font-size-adjust trong CSS, bạn có thể điều chỉnh kích cỡ văn bản của một phần tử. Thuộc tính này giúp bạn điều chỉnh x-height để làm cho font dễ đọc hơn. Giá trị của thuộc tính là bất kỳ số nào. Ví dụ:

1. **<html>**
2. **<head>**
3. **</head>**
4. **<body>**
5. **<p** style="font-size-adjust:0.61;"**>**
6. Doan van nay su dung thuoc tinh font-size-adjust co gia tri 0.61.
7. **</p>**
8. **</body>**
9. **</html>**

*Thiết lập Font Size bằng Pixel trong CSS*

Việc thiết lập kích cỡ font bằng Pixel giúp bạn kiểm soát hoàn toàn kích cỡ văn bản. Ví dụ minh họa:

1. h1 {
2. **font-size**: 40px;
3. }
4. h2 {
5. **font-size**: 30px;
6. }
7. p {
8. **font-size**: 14px;
9. }

Ghi chú: Nếu bạn sử dụng Pixel, bạn vẫn có thể sử dụng tính năng Zoom để tăng giảm kích cỡ của cả trang web.

*Thiết lập Font Size bằng Em trong CSS*

Để cho phép người dùng tăng giảm kích cỡ của văn bản (trong menu của trình duyệt), nhiều lập trình viên sử dụng đơn vị em thay cho pixel.

Đơn vị em được đề nghị sử dụng bởi W3C.

1em thì bằng với kích cỡ font hiện tại. Kích cỡ văn bản mặc định là 16px, do đó kích cỡ mặc định của 1em sẽ là 16px.

*Thuộc tính font trong CSS*

-Để tối thiểu hóa lượng code cần phải viết, bạn có thể sử dụng thuộc tính font trong CSS để thiết lập tất cả thuộc tính liên quan tới Font đã được trình bày ở phần trên.

-Trong thuộc tính font này, bạn thiết lập các thuộc tính liên quan tới font theo thứ tự như trong cú pháp sau:

font-style font-variant font-weight font-size/line-height font-family

-Trong đó, giá trị của hai thuộc tính font-size và font-family là bắt buộc. Nếu bạn không xác định một trong các giá trị khác thì giá trị mặc định sẽ được sử dụng (nếu có).

Ghi chú: thuộc tính line-height trong CSS được sử dụng để thiết lập khoảng cách giữa các dòng.

Ví dụ đơn giản về cách sử dụng thuộc tính font trong CSS:

1. **<html>**
2. **<head>**
3. **</head>**
4. **<body>**
5. **<p** style="font:italic small-caps bold 15px georgia;"**>**
6. Bang Style Rule tren, ban co the xac dinh gia tri cho tat ca cac thuoc tinh lien quan toi font.
7. **</p>**
8. **</body>**
9. **</html>**

**2.3.1.15 CSS icons**

**Cách thêm biểu tượng**

Cách đơn giản nhất để thêm biểu tượng vào trang HTML của bạn là với thư viện biểu tượng, chẳng hạn như Font Awesome.

Thêm tên của lớp biểu tượng đã chỉ định vào bất kỳ thành phần HTML nội tuyến nào (như <i> hoặc <span>).

Tất cả các biểu tượng trong các thư viện biểu tượng bên dưới, là các vectơ có thể mở rộng có thể được tùy chỉnh bằng CSS (kích thước, màu sắc, bóng, v.v.)

**Font Awesome Icons**

Để sử dụng các biểu tượng Font Awesome, hãy truy cập fontawgie.com, đăng nhập và nhận mã để thêm vào phần <head> trên trang HTML của bạn:

<scriptsrc = "https://kit.fontawclaw.com/yourcode.js"></ script>

Đọc thêm về cách bắt đầu với Font Awesome trong hướng dẫn Font Awesome 5 của chúng tôi.

Lưu ý: Không cần tải xuống hoặc cài đặt

Vi du :

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<script** src="https://kit.fontawesome.com/a076d05399.js"**></script>**
5. **</head>**
6. **<body>**
8. **<i** class="fas fa-cloud"**></i>**
9. **<i** class="fas fa-heart"**></i>**
10. **<i** class="fas fa-car"**></i>**
11. **<i** class="fas fa-file"**></i>**
12. **<i** class="fas fa-bars"**></i>** **</body>**
13. **</html>**

**Bootstrap Icons**

Để sử dụng glyphicons Bootstrap, hãy thêm dòng sau vào phần <head> trên trang HTML của bạn:

**<link** rel = "bản định kiểu" href = "https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"**>**

Lưu ý: Không cần tải xuống hoặc cài đặt!

Vd:

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<link** rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"**>**
5. **</head>**
6. **<body>**
8. **<i** class="glyphicon glyphicon-cloud"**></i>**
9. **<i** class="glyphicon glyphicon-remove"**></i>**
10. **<i** class="glyphicon glyphicon-user"**></i>**
11. **<i** class="glyphicon glyphicon-envelope"**></i>**
12. **<i** class="glyphicon glyphicon-thumbs-up"**></i>**
14. **</body>**
15. **</html>**

**Google Icons**

Để sử dụng các biểu tượng của Google, hãy thêm dòng sau vào phần <head> trên trang HTML của bạn:

<link rel = "bản định kiểu" href = "https://fonts.googleapis.com/icon?family=M vật liệu + Ention">

Lưu ý: Không cần tải xuống hoặc cài đặt!

ví dụ

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<link** rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons"**>**
5. **</head>**
6. **<body>**
8. **<i** class="material-icons"**>**cloud**</i>**
9. **<i** class="material-icons"**>**favorite**</i>**
10. **<i** class="material-icons"**>**attachment**</i>**
11. **<i** class="material-icons"**>**computer**</i>**
12. **<i** class="material-icons"**>**traffic**</i>**
14. **</body>**
15. **</html>**

**2.3.1.16 CSS links**

Ngoài những cách thông thường để tạo và trang trí đường link mà bạn đã biết trong HTML, thì trong CSS, bạn sẽ có các thuộc tính đa dạng với nhiều giá trị để giúp được link trong Webpage của bạn trở nên đẹp và chuyên nghiệp hơn.

Tạo link style trong CSS

Để tạo Link Style, chẳng hạn như tạo màu, font, … cho các đường link trong CSS, bạn có thể sử dụng nhiều thuộc tính CSS đa dạng, ví dụ như color, font-family, background, …

vd

a {

color:hotpink;

}

Ngoài ra, để thiết lập các trạng thái khác nhau cho link, bạn có thể sử dụng các thuộc tính:

• Trạng thái :link – biểu thị rằng trang web này trình duyệt chưa lưu (tức người sử dụng lần đầu tiên click vào đường dẫn này).

• Trạng thái :visited – biểu thị rằng đường dẫn tới trang web này đã được lưu bởi trình duyệt (tức là người sử dụng đã click vào đường dẫn này trước đó rồi).

• Trạng thái :hover – biểu thị rằng khi người sử dụng di chuyển chuột qua phần tử mà chứa link đó (tức là phần tử đó là một link khi người sử dụng di chuyển chuột qua phần tử đó).

• Trạng thái :active – biểu thị đường link là active khi người sử dụng click chuột vào.

Ghi chú:

Trạng thái :hover PHẢI theo sau các trạng thái :link và :visited trong định nghĩa CSS thì mới có hoạt động. Nếu thiếu một trong hai trạng thái trên, thì các Style Rule bạn đã định nghĩa cho :hover sẽ không có giá trị.

Trạng thái :active PHẢI theo sau trạng thái :hover trong định nghĩa CSS.

Ví dụ

1. **<style** type="text/css"**>**
2. a:link {color: #000000}
3. a:visited {color: #006600}
4. a:hover {color: #FFCC00}
5. a:active {color: #FF00CC}
6. **</style>**

Tiếp theo, chúng ta sẽ cùng tìm hiểu cách sử dụng các thuộc tính này để tạo các hiệu ứng khác nhau cho đường link trong CSS.

Thiết lập màu cho Link trong CSS

Để thiết lập màu cho Link, bạn sử dụng thuộc tính color trong CSS. Như đã trình bày trong chương Màu trong CSS, giá trị của thuộc tính color này có thể là tên màu hoặc một giá trị Hex Code, …

1. **<html>**
2. **<head>**
3. **<style** type="text/css"**>**
4. a:link {color:#000000}
5. **</style>**
6. **</head>**
7. **<body>**
8. **<a** href=""**>**Click vao link nay**</a>**
9. **</body>**
10. **</html>**

*Thiết lập màu cho Visited Link trong CSS*

Trạng thái :visited chỉ rằng đường link này đã được lưu bởi trình duyệt. Để thiết lập màu cho loại đường này, bạn theo dõi ví dụ sau:

1. **<html>**
2. **<head>**
3. **<style** type="text/css"**>**
4. a:visited {color: #006600}
5. **</style>**
6. **</head>**
7. **<body>**
8. **<a** href=""**>** Click vao link nay**</a>**
9. **</body>**
10. **</html>**

*Thay đổi màu cho Link khi Di chuyển chuột qua*

Với trường hợp này, bạn sử dụng trạng thái :hover. Nên ghi nhớ rằng, trong phần định nghĩa CSS, để đường Link này có thể hoạt động, bạn nên đặt trạng thái này sau hai trạng thái :link và :visited. Tuy nhiên, để cho đơn giản thì trong ví dụ sau mình thực hiện như sau:

1. **<html>**
2. **<head>**
3. **<style** type="text/css"**>**
4. a:hover {color: #FFCC00}
5. **</style>**
6. **</head>**
7. **<body>**
8. **<a** href=""**>**Click vao link nay**</a>**
9. **</body>**
10. **</html>**

Khi bạn di chuyển chuột qua đường link trên, màu link sẽ chuyển sang màu vàng.

Thay đổi màu cho Actived Link trong CSS

Để biểu thị một link nào đó đang active, bạn có thể thay đổi màu của link đó sang một màu khác chẳng hạn. Để thiết lập trạng thái này, bạn sử dụng :active trong CSS. Ví dụ sau minh họa cách thay đổi màu cho Actived Link bởi sử dụng thuộc tính color trong CSS.

1. **<html>**
2. **<head>**
3. **<style** type="text/css"**>**
4. a:active {color: #FF00CC}
5. **</style>**
6. **</head>**
7. **<body>**
8. **<a** href=""**>**Click vao link nay**</a>**
9. **</body>**
10. **</html>**

Khi bạn click chuột vào link này, màu của link sẽ chuyển sang màu hồng.

Sử dụng thuộc tính text-decoration với Link trong CSS

Nếu bạn để ý các ví dụ trên, bạn sẽ thấy rằng các đường link trên đều có dấu gạch chân bên dưới. Vậy làm cách nào để làm biến mất dấu gạch chân này cho đường link, bạn sử dụng thuộc tính text-decoration với giá trị là none. (Bạn tham khảo chương Text trong CSS để có các giá trị cho thuộc tính này)

Ví dụ

1. **<html>**
2. **<head>**
3. **<style>**
4. a:link {
5. text-decoration: none;
6. }
8. a:visited {
9. text-decoration: none;
10. }
12. a:hover {
13. text-decoration: underline;
14. }
16. a:active {
17. text-decoration: underline;
18. }
19. **</style>**
20. **</head>**
21. **<body>**
23. **<p><b><a** href="default.asp" target="\_blank"**>**This is a link**</a></b></p>**
24. **<p><b>**Note:**</b>** a:hover MUST come after a:link and a:visited in the CSS definition in order to be effective.**</p>**
25. **<p><b>**Note:**</b>** a:active MUST come after a:hover in the CSS definition in order to be effective.**</p>**
27. **</body>**
28. **</html>**

*Thiết lập màu nền cho Link trong CSS*

Để thiết lập màu nền cho Link, bạn sử dụng thuộc tính background-color trong CSS. Để tìm hiểu các giá trị của thuộc tính này, bạn tham khảo chương Color trong CSS.

Ví dụ

1. **<html>**
2. **<head>**
3. **<style>**
4. a:link {
5. background-color: yellow;
6. }
8. a:visited {
9. background-color: cyan;
10. }
12. a:hover {
13. background-color: lightgreen;
14. }
16. a:active {
17. background-color: hotpink;
18. }
19. **</style>**
20. **</head>**
21. **<body>**
23. **<p><b><a** href="default.asp" target="\_blank"**>**This is a link**</a></b></p>**
24. **<p><b>**Note:**</b>** a:hover MUST come after a:link and a:visited in the CSS definition in order to be effective.**</p>**
25. **<p><b>**Note:**</b>** a:active MUST come after a:hover in the CSS definition in order to be effective.**</p>**
27. **</body>**
28. **</html>**

Ngoài ra, bạn có thể kết hợp nhiều thuộc tính khác nhau trong CSS để làm cho Link của bạn giống như một Box, hoặc một Button.

Sử dụng ví dụ trên, bạn chỉ cần thay Style Rule:

1. a:link, a:visited {
2. **background-color**: #f44336;
3. **color**: white;
4. **padding**: 14px 25px;
5. **text-align**: center;
6. **text-decoration**: none;
7. **display**: inline-block;
8. }
10. a:hover, a:active {
11. **background-color**: red;
12. }

***2.3.1.17 CSS lits***

Trình bày hay liệt kê một danh sách có kèm theo các bullet hoặc dưới dạng đánh số giúp phần văn bản của bạn dễ đọc và tạo cảm giác thân thiện hơn. Để tạo style riêng cho phần list, bạn có thể sử dụng các thuộc tính trong CSS.

*Có 5 thuộc tính trong CSS*

· Thuộc tính list-style-type cho phép bạn kiểm soát hình dạng hay bề ngoài của Marker (giống như Bullet) chẳng hạn ở dạng hình tròn, hình vuông, hay dạng số, ...

· Thuộc tính list-style-position xác định rằng marker nên xuất hiện bên trong hay bên ngoài luồng hiển thị nội dung.

· Thuộc tính list-style-image: sử dụng một hình ảnh để làm marker thay cho các bullet hoặc số.

· Thuộc tính list-style: sử dụng thuộc tính này, bạn có thể khai báo một lần mà vẫn có thể xác định được tất cả các thuộc tính trên.

· Thuộc tính marker-offset xác định khoảng cách giữa một marker và phần text trong danh sách.

## Thuộc tính list-style-type trong CSS

· Để điều khiển hình dạng (hình tròn, hình vuông, …) của bullet trong trường hợp một danh sách không có thứ tự và style của các ký tự số (số La Mã, số tự nhiên, …) trong trường hợp danh sách đã được sắp xếp, bạn sử dụng thuộc tính list-style-type trong CSS.

· Dưới đây là các giá trị có thể được sử dụng cho dạng danh sách không có thứ tự:

|  |  |
| --- | --- |
| ***Giá trị*** | ***Miêu tả*** |
| *none* | *Không hiển thị* |
| *disc (default)* | *Một dấu chấm tròn* |
| *circle* | *Một vòng tròn trống* |
| *square* | *Một hình vuông* |

*Dưới đây là các giá trị có thể được sử dụng cho dạng danh sách đã qua sắp xếp:*

|  |  |  |
| --- | --- | --- |
| ***Giá trị*** | ***Miêu tả*** | ***Ví dụ*** |
| *decimal* | *Số tự nhiên* | *1,2,3,4,5* |
| *decimal-leading-zero* | *Dạng số, bắt đầu từ 01* | *01, 02, 03, 04, 05* |
| *lower-alpha* | *Dạng chữ cái thường* | *a, b, c, d, e* |
| *upper-alpha* | *Dạng chữ cái hoa* | *A, B, C, D, E* |
| *lower-roman* | *Dạng số La Mã thường* | *i, ii, iii, iv, v* |
| *upper-roman* | *Dạng số La Mã hoa* | *I, II, III, IV, V* |
| *lower-greek* | *Dạng chữ Hy Lạp thường* | *alpha, beta, gamma* |

## Thuộc tính list-style-position trong CSS

Để xác định xem marker nên hiển thị ở bên trong hay bên ngoài luồng hiển thị nội dung, bạn sử dụng thuộc tính list-style-position trong CSS.

Thuộc tính này có thể nhận giá trị: none, inside hoặc outside.

## Thuộc tính list-style-image trong CSS

Nếu bạn muốn tạo điểm nhấn cho phần danh sách bạn đang tạo, bạn có thể sử dụng một hình ảnh đẹp nào đó để sử dụng thay cho các marker đơn điệu có sẵn. Để thực hiện điều này, bạn sử dụng thuộc tính list-style-image trong CSS.

Cú pháp của thuộc tính này tương tự như thuộc tính background-image, tại đây, bạn cần xác định URL tới nơi bạn lưu giữ hình ảnh. Lưu ý rằng, nếu không tìm thấy hình ảnh, thì các bullet mặc định sẽ được sử dụng.

## Thuộc tính list-style trong CSS

Sử dụng thuộc tính list-style, bạn có thể xác định tất cả style cho danh sách (toàn bộ các thuộc tính ở trên) chỉ trong một khái báo Style Rule. Thứ tự của giá trị các thuộc tính này có thể là tùy ý.

## Thuộc tính marker-offset trong CSS

Nếu bạn cảm thấy khoảng cách mặc định giữa marker và phần nội dung hiển thị là dài hơn hoặc ngắn hơn bạn mong muốn, và bạn muốn thiết lập lại, bạn nên sử dụng thuộc tính marker-offset. Giá trị của thuộc tính này ở dưới dạng độ dài (đơn vị px, pt, cm, …).

Ghi chú: IE 6 và Netscape 7 không hỗ trợ thuộc tính này.

## Tạo màu cho List trong CSS

Với các thuộc tính đã học từ các chương trước, bạn có thể sử dụng kết hợp chúng với nhau để tạo một danh sách đẹp và lộng lẫy hơn bằng cách thêm màu nền chẳng hạn.

***2.3.1.18 CSS tables***

**Bảng trong CSS**

Bảng là một công cụ hiển thị dữ liệu rõ ràng và hiệu quả. Mặc dù với các công nghệ mới (như Kendo UI), thì việc hiển thị dữ liệu bằng các thẻ div thường được sử dụng hơn. Tuy nhiên, với các ứng dụng Webpage nhỏ và với lượng dữ liệu hiển thị là không lớn thì sử dụng bảng là khá tiện lợi. Chương này sẽ trình bày cách sử dụng các thuộc tính khác nhau trong CSS để làm cho bảng của bạn đẹp hơn.

*Dưới đây là một số thuộc tính trong CSS:*

Thuộc tính border được sử dụng để thiết lập đường viền cho bảng.

Thuộc tính border-collapse xác định rằng các đường viền của bảng nên được vào hợp thành một đường viền.

Thuộc tính caption-side được sử dụng trong phần tử <caption>. Theo mặc định, chúng sẽ được hiển thị ở phần bên trên của bảng. Sử dụng thuộc tính này, bạn có thể xác định vị trí hiển thị của phần tử caption này.

Thuộc tính empty-cells xác định xem có hiển thị đường viền không nếu một ô là trống.

Thuộc tính table-layout cho phép bạn thiết lập layout cho bảng.

Độ rộng và chiều cao bảng trong CSS

Để xác định độ rộng và chiều cao cho bảng, bạn sử dụng hai thuộc tính width và height trong CSS. Hai thuộc tính này có thể nhận các giá trị là % hoặc px.

*Căn chỉnh bảng trong CSS*

-Căn chỉnh bảng theo chiều ngang trong CSS

Bạn sử dụng thuộc tính text-align để xác định sự căn chỉnh nội dung theo chiều ngang của bảng. Thuộc tính này có thể nhận các giá trị left, right hoặc center. Ví dụ:

1. th {
2. **text-align**: left;
3. }

*Căn chỉnh bảng theo chiều dọc trong CSS*

Để căn chỉnh bảng theo chiều dọc, bạn sử dụng thuộc tính vertical-align có thể nhận các giá trị top, bottom, middle. Theo mặc định thì nội dung của bảng có căn chỉnh dọc với giá trị middle. Ví dụ:

1. td {
2. **height**: 50px;
3. **vertical-align**: bottom;
4. }

*Thuộc tính padding trong CSS*

Để điều khiển khoảng cách giữa đường viền và nội dung của bảng, bạn sử dụng thuộc tính padding trong CSS. Giá trị có thể nhận của thuộc tính này là ở dưới dạng đơn vị px.

Ví dụ

1. th, td {
2. **padding**: 15px;
3. **text-align**: left;
4. }

*Thuộc tính border-spacing trong CSS*

Thuộc tính border-spacing xác định khoảng cách giữa các đường viền của các ô trong bảng. Thuộc tính này có thể nhận một hoặc hai giá trị (có đơn vị là độ dài).

Nếu cung cấp một giá trị, thì giá trị này sẽ được áp dụng cho đường viền ngang và dọc. Nếu cung cấp hai giá trị, thì tương ứng theo thứ tự sẽ áp dụng cho đường viền ngang và đường viền dọc.

***2.3.1.19 CSS displays***

Định nghĩa và sử dụng

Thuộc tính display xác định loại hiển thị của thành phần.

Cấu trúc

tag {

display: giá trị;

}

Với giá trị như sau:



***2.3.1.20 CSS max-width***

**Khái niệm, định nghĩa**

Mặc định các khối phần tử trong HTML sẽ có chiều rộng tối đa (full viền, trải dài từ trái sang phải). Tuy nhiên, trong một số trường hợp, ta cần phải giới hạn chiều rộng (theo px, em hoặc %) cho một số phần tử để nó hiển thị đúng theo yêu cầu thiết kế , ta phải dùng tới thuộc tính max-width. Max-width trong CSS dùng để thiết lập chiều rộng tối đa cho một phần tử.

Trong đó, tag là tất cả phần tử HTML, ngoài trừ các phần tử table. Các giá trị có thể sử dụng cho thuộc tính max-width như sau:

• Đơn vị: (max-width: 50px): chiều rộng tính từ mép ngoài cùng bên trái của phần tử, đơn vị có thể là px, em hoặc %,…

• None: (max-width: none): không sử dụng chiều rộng lớn nhất, đây là dạng mặc định của phần tử.

• Inherit: (max-width: inherit): xác định thừa hưởng thuộc tính từ phần tử cha (phần tử bao ngoài).

Ví dụ thuộc tính max-width trong CSS

Bạn có thể theo dõi ví dụ sau đây để hiểu cách hoạt động của thuộc tính max-width trong CSS:

Ví dụ 1: Đặt giá trị max-width: 500px cho phần tử.

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<style>**
5. div.ex2 {
6. max-width:500px;
7. margin: auto;
8. border: 3px solid #73AD21;
9. }
10. **</style>**
11. **</head>**
12. **<body>**
13. **<div** class="ex2"**>**Đoạn này có chiều rộng tối đa là 500px và có bo viền độ dày 3px**</div>**
14. **</body>**
15. **</html>**

**Điều kiện sử dụng**

Thuộc tính max-width sử dụng được trên các trình duyệt Opera, Chrome, Cốc Cốc, IE (ngoại trừ IE6; IE8 vẫn dùng được nhưng phải có !Doctype.

Khi đã sử dụng thuộc tính max-width, chiều rộng của box sẽ không vượt qua giá trị của max-width.

Thật dễ hiểu phải không nào, cái này rất quan trọng để sau này ta có thể áp dụng vào Responsive cho Website nên các bạn bạn cần phải nhắm chắc và hiểu rõ bản chất của thuộc tính này…

***2.3.1.21 CSS Position***

**Định nghĩa và sử dụng**

Thuộc tính position xác định loại của phương pháp định vị trí cho thành phần.

Thuộc tính position thường dùng kèm với các thuộc tính định vị trí: left, right, bottom, top

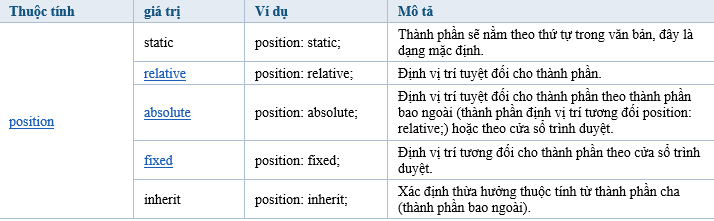
Cấu trúc

tag {

position: giá trị;

}

Với giá trị như sau:

**

VD:

1. div {
2. **border**: 1px solid red;
3. **height**: 80px;
4. **position**: relative;
5. }
7. div p {
8. **background**: #00CCFF;
9. **width**: 80px;
10. **position**: absolute;
11. right: -20px;
12. top: 15px;
13. }

***2.3.1.22 CSS overflow***

Định nghĩa và sử dụng

Thuộc tính overflow xác định điều gì sẽ xảy ra nếu một thành phần box tràn nội dung.

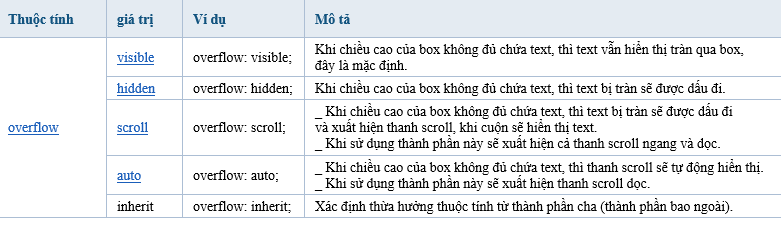
Cấu trúc

tag {

overflow: giá trị;

}

Với giá trị như sau:



VD:

p {

border: 1px solid red;

height: 100px;

overflow: hidden;

}

***2.3.1.23 CSS float***

Định nghĩa và sử dụng

Thuộc tính float xác định có hay không một thành phần được float (trôi nổi).

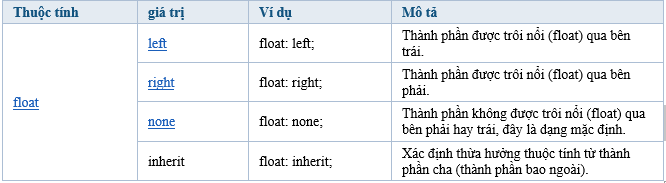
Cấu trúc

tag {

float: giá trị;

}

Với giá trị như sau:



***2.3.1.24 CSS inline-block***

Có thể nhận thấy rằng, một số thẻ HTML như <div>, <p>, <ul> luôn bắt đầu bằng một dòng mới và có độ dài dòng trải dài 100% trang web, trong khi các thẻ HTML khác như <span>, <img> hoặc <a> thì thường được đặt nối tiếp nhau trên cùng một dòng. Tại sao lại có sự khác biệt như vậy? Hãy đi tìm câu trả lời qua bài học về giá trị của thuộc tính display ngay sau đây.

Trả lời một cách đơn giản, sự khác biệt này là do giá trị hiển thị khác nhau: Block và Inline.

**Các giá trị của thuộc tính Display**

***Giá trị display: inline***

Inline là cách hiển thị trên một hàng và chiều rộng của thẻ đó sẽ phụ thuộc vào nội dung bên trong của thẻ, vì vậy nhiều thẻ có thể nằm trên cùng một hàng. Tuy nhiên inline không thể sử dụng được các thuộc tính width, height và top-bottom margin. Các thẻ HTML được hiển thị mặc định inline là <span>, <a>, <strong>, <b>, <i>...

***Giá trị display: block***

Block là cách hiển thị luôn bắt đầu bằng một dòng mới và có chiều rộng bằng 100% chiều rộng trang web. Vì vậy khi bạn dùng thẻ này thì mặc dù nội dung ngắn nhưng các thẻ khác ở phía dưới vẫn được nằm ở một dòng khác tách biệt. Khác biệt của thẻ này là dùng được thuộc tính width và height. Các thẻ HTML hiển thị mặc đinh block là: <div>, <p>, <h1> đến <h6>, <header>, <footer>, <section>, <nav>..

***Giá trị display: inline-block***

Inline-block là cách hiển thị kết hợp cả hai cách trên, chuyển phần tử về hiển thị trên cùng một hàng nhưng nó vẫn thừa hưởng các đặc tính của block.

So với display: inline, cách hiển thị inline-block có thể sử dụng được các thuộc tính width, height và top-bottom margin.

So với display: block, cách hiển thị inline-block không làm ngắt dòng sau khi kết thúc thẻ, do đó các thẻ có thế được đặt nằm cạnh nhau

**Sử dụng inline-block để tạo liên kết điều hướng**

Display: inline-block được sử dụng chủ yếu cho việc hiển thị danh sách điều hướng theo chiều ngang:

1. .nav {
2. **background-color**: yellow;
3. **list-style-type**: none;
4. **text-align**: center;
5. **padding**: 0;
6. **margin**: 0;
7. }
9. .nav li {
10. **display**: inline-block;
11. **font-size**: 20px;
12. **padding**: 20px;
13. }

Theo mặc định, các mục danh sách được hiển thị theo chiều dọc. Ở ví dụ này, ta sử dụng display: inline-block để hiển thị chúng theo chiều ngang (cạnh nhau).

Lưu ý: Nếu bạn thay đổi kích thước cửa sổ trình duyệt, các liên kết sẽ tự động ngắt xuống dòng khi nội dung dài quá chiều rộng web.

***2.3.1.25 CSS align***

Align là sự thiết lập vị trí của phần tử hoặc nội dung của phần tử. Trong CSS có rất nhiều thuộc tính cho phép căn chỉnh (align) một phần tử như margin, padding, text-align, position, float... Align giúp ích cho bạn rất nhiều trong quá trình định dạng văn bản, hình ảnh.

**Căn giữa cho phần tử - Sử dụng margin:auto**

Trong CSS, để căn giữa một phần tử khối (như <div>), ta sử dụng thuộc tính margin có giá trị là auto để căn chỉnh phần tử nằm ở giữa so với phần tử cha của nó.

Phần tử sau khi chiếm không gian nhất định, khoảng trống còn lại sẽ được chia đều cho hai bên lề

Lưu ý:

• Căn chỉnh giữa sẽ không tạo ra hiệu ứng nếu không có thuộc tính width (hoặc width có giá trị là 100%) vì nó chiếm hết động rộng sẵn có, không thể căn giữa được nữa.

• Thuộc tính margin: auto chỉ căn giữa cho phần tử block như <div>, <p>, <h1> đến <h6>, <header>, <footer>, <section>, <nav>... còn các phần tử inline như <span>, <a>, <strong>, <b>, <i>... sẽ không được áp dụng.

• Có thể ghi đè lại phần tử hiển thị block hay inline bằng cách sử dụng thuộc tính display. Ví dụ: span { display: block;}.

Căn giữa hình ảnh

• Để căn giữa một ảnh, đặt margin-left và margin-right thành auto và làm cho hình ảnh trở thành phần tử dạng block như đã nói ở lưu ý phía trên.

**Căn chỉnh văn bản theo chiều ngang - Sử dụng text-align**

Trong CSS có thuộc tính text-align cho phép căn chỉnh văn bản nội dung sẽ nằm bên trái, ở giữa hay bên phải của phần tử bằng các giá trị:

• text-align: center - căn giữa

• text-align: left - căn trái

• text-align: right - căn phải

• Lưu ý: Thuộc tính text-align cũng chỉ áp dụng cho phần tử block vì inline chỉ chiếm độ rộng vừa đủ nội dung của nó.

• Tìm hiểu thêm về định dạng văn bản trong CSS tại đây

**Căn trái/phải - Sử dụng position**

Một phương pháp khác để căn chỉnh phần tử là sử dụng position: absolute.

Chú ý: Nếu một phần tử đang được float cao hơn phần tử cha thì sẽ gây ra hiện tượng tràn nội dung ra bên ngoài. Để khắc phục, ta sử dụng clearfix overflow: auto

Căn giữa theo chiều dọc - Sử dụng padding

Có nhiều cách để căn giữa một phần tử theo chiều dọc trong CSS. Giải pháp đơn giản nhất đó là sử dụng padding.

1. .center {
2. **padding**: 70px 0;
3. **border**: 3px solid green;

**Căn giữa theo chiều dọc - Sử dụng line-height**

Bạn cũng có cách khác để căn giữa theo chiều dọc là sử dụng thuộc tính line-height với giá trị giống như thuộc tính height.

Căn giữa theo chiều dọc - Sử dụng position và transform

Nếu không sử dụng padding và line-height như trên, bạn có thể dùng cách thứ ba là sử dụng position và transform

Lưu ý: Thuộc tính transform không được hỗ trợ trong IE8 và các phiên bản trước đó

***2.3.1.26 CSS combinators***

*Combinator (tổ hợp)* thể hiện mối quan hệ giữa các selector, cho phép kết hợp các selector lại với nhau dưới dạng các chuỗi ký tự.

Trong CSS, có 4 combinator quen thuộc như sau:

· *Descendant combinator (space):* Select những phần tử là con cháu của một phần tử khác.

*· Child combinator (>):* Select những phần tử là con của một phần tử khác.

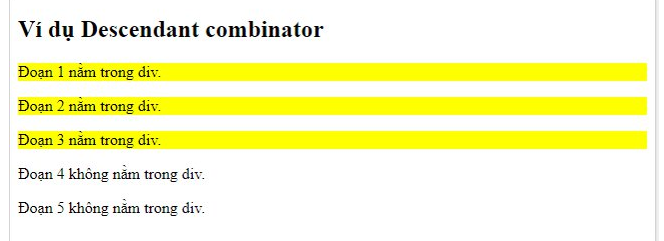
Chú ý ở đây "con cháu" được dùng để chỉ phần tử nằm trong phần tử khác, còn "con" là chỉ phần tử nằm trực tiếp ngay bên trong của phần tử khác. Ví dụ như <div><span><p>example</p><span></div> thì chỉ có span là "con" của div, còn cả span và p đều là "con cháu" của div.

*· Adjacent sibling combinator (+):* Select phần tử nằm liền kề và ngang hàng với phần tử khác (các phần tử "anh em" kề nhau).

· *General sibling combinator (~):* Select phần tử ngang hàng với phần tử khác (các phần tử "anh em").

*Descendant combinator*

*Descendant combinator (space)* cho phép kết hợp những phần tử là con cháu của một phần tử khác.

**Ví dụ:**

## Child combinator

*Child combinator (>)* cho phép kết hợp những phần tử là con "trực tiếp" của một phần tử khác (như đã giải thích ở trên).

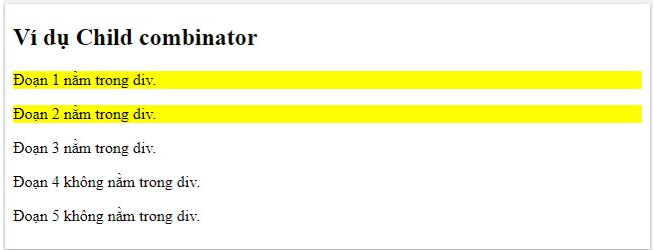
div > p {

background-color: yellow;

}

Ở đây, phần tử con là thẻ p, phần tử thứ nhất là div. Thẻ p nằm ở trong thẻ div sẽ có background-color: yellow, còn các thẻ khác nằm trong p thì không.

**Ví dụ:**

****

## Adjacent sibling combinator

*Adjacent sibling combinator (+)* cho phép chọn những phần tử nằm liền kề và ngang hàng với phần tử được chỉ định (các phần tử "anh em" kề nhau).

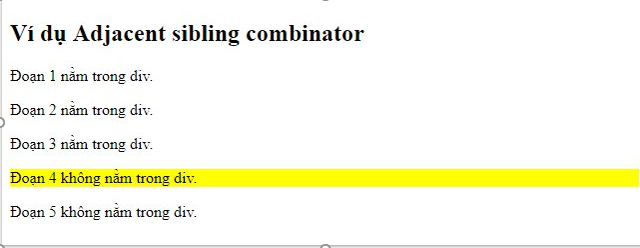
div + p {

background-color: yellow;

}

Ở đây, phần tử p nằm liền kề và ngang hàng với div sẽ có background-color: yellow, còn các thẻ p khác nằm trong div hoặc thẻ p ngang hàng nhưng không liền kề với div thì không.

**Ví dụ:**

****

## General sibling combinator

*General sibling combinator (~)* cho phép chọn những phần tử nằm ngang hàng với phần tử được chỉ định (các phần tử "anh em").

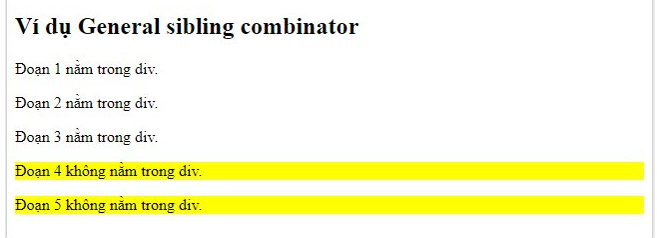
div ~ p {

background-color:yellow;

}

Ở đây, phần tử p nằm ngang hàng với div sẽ có background-color: yellow, còn các thẻ p khác nằm trong div thì không.

***Ví dụ:***

******

***2.3.1.27 CSS Pseudo-class***

* Giả lớp là gì?

Một pseudo-class được sử dụng để xác định tình trạng đặc biệt của một phần tử.

Nó có thể được sử dụng để :

* Định dạng một phần tử khi người dùng lướt chuột qua nó.
* Kiểu truy cập và liên kết không mong muốn khác.
* Định dnagj một phần tử khi nó được làm nổi bật.
* Cú pháp

selector:pseudo-class {  
  property:value;  
}

* Mấu neo Pseudo-classes

Liên kết có thể được hiển thị theo những cách khác nhau :

/\* unvisited link \*/  
a:link {  
  color: #FF0000;  
}  
  
/\* visited link \*/  
a:visited {  
  color: #00FF00;  
}  
  
/\* mouse over link \*/  
a:hover {  
  color: #FF00FF;  
}  
  
/\* selected link \*/  
a:active {  
  color: #0000FF;  
}

Chú ý: a:hover phải sau a:limk và a:visited trong định nghĩa CSS để có hiệu quả! a:active phải sau a:hover trong định nghĩa CSS để có hiệu quả!

* Giả lớp và lớp CSS

Giả lớp có thể được kết hợp với các lớp CSS:

Khi bạn di chuột qua liên kết trong ví dụ này, nó sẽ thay đổi màu sắc:

a.highlight:hover {  
  color: #ff0000;  
}

* Hover on <div>

Ví dụ sử dụng :hover pseu-class trên một phần tử <div>:

div:hover {  
  background-color: blue;  
}

* Đơn giản Tooltip Hover

Di chuột qua một phần tử <div> để hiển thị một phần tử <p> (như một tooltip):

p {  
  display: none;  
  background-color: yellow;  
  padding: 20px;  
}  
  
div:hover p {  
  display: block;  
}

* CSS – The:first-child Pseudo-class

:first-child pseudo-class so khớp với một phần tử là đứa con đầu của phần tử khác.

* So khớp với phần tử <p> đầu tiên

Trong ví dụ sau, bộ chọn phù hợp với bất kỳ phần tử <p> đó là đứa con đầu của bất kỳ phần tử nào:

p:first-child {  
  color: blue;  
}

* So khớp tất cả phần tử <i> trong tất cả phần tử con đầu tiên <p>

p i:first-child {  
  color: blue;  
}

* CSS – The :lang Pseudo-class

:lang pseudo-class cho phép bạn xác định quy tắc đặc biệt cho các ngôn ngữ khác nhau.

Trong ví dụ dưới đây, :lang xác định các dấu ngoặc kép cho phần tử <q> với lang=”no”:

<html>  
<head>  
<style>  
q:lang(no) {  
  quotes: "~" "~";  
}  
</style>  
</head>  
<body>  
  
<p>Some text <q lang="no">A quote in a paragraph</q> Some text.</p>  
  
</body>  
</html>

* Pseudo classes trong CSS

|  |  |  |
| --- | --- | --- |
| **Selector** | **Ví dụ** | **Mô tả ví dụ** |
| [:active](https://www.w3schools.com/cssref/sel_active.asp) | a:active | Chọn liên kết đang hoạt động. |
| [:checked](https://www.w3schools.com/cssref/sel_checked.asp) | input:checked | Chọn mỗi phần tử <input> đã kiểm tra. |
| [:disabled](https://www.w3schools.com/cssref/sel_disabled.asp) | input:disabled | Chọn mỗi phần tử <input> bị vô hiệu hóa. |
| [:empty](https://www.w3schools.com/cssref/sel_empty.asp) | p:empty | Chọn mỗi phần tử <p> không có con. |
| [:enabled](https://www.w3schools.com/cssref/sel_enabled.asp) | input:enabled | Chọn mỗi phần tử <input> được bật. |
| [:first-child](https://www.w3schools.com/cssref/sel_firstchild.asp) | p:first-child | Chọn tất cả các phần tử <p> đó là con đầu tiên của cha mẹ nó. |
| [:first-of-type](https://www.w3schools.com/cssref/sel_first-of-type.asp) | p:first-of-type | Chọn mỗi phần tử <p> là phần tử <p> đầu tiên của cha / mẹ. |
| [:focus](https://www.w3schools.com/cssref/sel_focus.asp) | input:focus | Chọn phần tử <input> cần nhấn mạnh. |
| [:hover](https://www.w3schools.com/cssref/sel_hover.asp) | a:hover | Chọn liên kết khi di chuột qua. |
| [:in-range](https://www.w3schools.com/cssref/sel_in-range.asp) | input:in-range | Chọn phần tử <input> với một giá trị trong phạm vi được chỉ định. |
| [:invalid](https://www.w3schools.com/cssref/sel_invalid.asp) | input:invalid | Chọn tất cả các phần tử <input> có giá trị không hợp lệ. |
| [:lang(*language*)](https://www.w3schools.com/cssref/sel_lang.asp) | p:lang(it) | Chọn mỗi phần tử <p> với một giá trị thuộc tính lang bắt đầu với “it” |
| [:last-child](https://www.w3schools.com/cssref/sel_last-child.asp) | p:last-child | Chọn mỗi phần tử <p> là con cuối cùng của cha mẹ. |
| [:last-of-type](https://www.w3schools.com/cssref/sel_last-of-type.asp) | p:last-of-type | Chọn mỗi phần tử <p> là phần tử <p> cuối cùng của cha mẹ. |
| [:link](https://www.w3schools.com/cssref/sel_link.asp) | a:link | Chọn tất cả các liên kết chưa được click. |
| [:not(selector)](https://www.w3schools.com/cssref/sel_not.asp) | :not(p) | Chọn mọi phần tử không phải là một phần tử <p>. |
| [:nth-child(n)](https://www.w3schools.com/cssref/sel_nth-child.asp) | p:nth-child(2) | Chọn mỗi phần tử <p> là con thứ hai của cha mẹ. |
| [:nth-last-child(n)](https://www.w3schools.com/cssref/sel_nth-last-child.asp) | p:nth-last-child(2) | Chọn mỗi phần tử <p> là phần tử <p> thứ hai của cha/mẹ nó, kể từ con cuối cùng. |
| [:nth-last-of-type(n)](https://www.w3schools.com/cssref/sel_nth-last-of-type.asp) | p:nth-last-of-type(2) | Chọn mỗi phần tử <p> là phần tử <p> thứ hai của cha/mẹ nó, tính từ con cuối cùng. |
| [:nth-of-type(n)](https://www.w3schools.com/cssref/sel_nth-of-type.asp) | p:nth-of-type(2) | Chọn mỗi phần tử <p> là phần tử <p> thứ hai của cha/mẹ n |
| [:only-of-type](https://www.w3schools.com/cssref/sel_only-of-type.asp) | p:only-of-type | Chọn mỗi phần tử <p> là yếu tố <p> duy nhất của cha mẹ nó. |
| [:only-child](https://www.w3schools.com/cssref/sel_only-child.asp) | p:only-child | Chọn mỗi phần tử <p> là con duy nhất của cha mẹ của nó. |
| [:optional](https://www.w3schools.com/cssref/sel_optional.asp) | input:optional | Chọn các phần tử <input> không có thuộc tính “require” |
| [:out-of-range](https://www.w3schools.com/cssref/sel_out-of-range.asp) | input:out-of-range | Chọn các phần tử <input> với một giá trị bên ngoài phạm vi được chỉ định. |
| [:read-only](https://www.w3schools.com/cssref/sel_read-only.asp) | input:read-only | Chọn các phần tử <input> với thuộc tính “readonly” được chỉ định |
| [:read-write](https://www.w3schools.com/cssref/sel_read-write.asp) | input:read-write | Chọn các phần tử <input> mà không có thuộc tính “readonly” |
| [:required](https://www.w3schools.com/cssref/sel_required.asp) | input:required | Chọn các phần tử <input> với thuộc tính “required” được chỉ định |
| [:root](https://www.w3schools.com/cssref/sel_root.asp) | root | Chọn phần tử gốc của tài liệu. |
| [:target](https://www.w3schools.com/cssref/sel_target.asp) | #news:target | Chọn phần tử #news đang hoạt động hiện tại (nhấp vào URL có chứa tên anchor đó) |
| [:valid](https://www.w3schools.com/cssref/sel_valid.asp) | input:valid | Chọn tất cả các phần tử <input> với giá trị hợp lệ. |
| [:visited](https://www.w3schools.com/cssref/sel_visited.asp) | a:visited | Chọn tất cả liên kết đã truy cập |

* Các phần tử Pseudo trong CSS:

|  |  |  |
| --- | --- | --- |
| **Selector** | **Ví dụ** | **Mô tả ví dụ** |
| [::after](https://www.w3schools.com/cssref/sel_after.asp) | p::after | Chèn nội dung sau mỗi phần tử <p> |
| [::before](https://www.w3schools.com/cssref/sel_before.asp) | p::before | Chèn nội dung trước mỗi phần tử <p> |
| [::first-letter](https://www.w3schools.com/cssref/sel_firstletter.asp) | p::first-letter | Chọn chữ cái đầu tiên của mỗi phần tử <p> |
| [::first-line](https://www.w3schools.com/cssref/sel_firstline.asp) | p::first-line | Chọn dòng đầu tiên của mỗi phần tử <p> |
| [::selection](https://www.w3schools.com/cssref/sel_selection.asp) | p::selection | Chọn phần của một phần tử được chọn bởi người dùng. |

***2.3.1.28 CSS Pseudo-element***

* Pseudo-Elements là gì?

Một CSS pseudo-element được sử dụng để xác định kiểu các phần được chỉ định của một phần tử.

Nó có thể được sử dụng để:

* Tạo kiểu cho chữ cái đầu tiên, hoặc dòng của một phần tử
* Chèn nội dung trước , hoặc sau nội dung của một phần tử
* Cú pháp

selector::pseudo-element {  
  property:value;  
}

Chú ý các ký hiệu dấu hai chấm kép - ::first-line với :first-line

Dấu hai chấm đôi thay thế các ký hiệu đơn cho các phần tử giả trong CSS3.

Cú pháp dấu hai chấm đơn được sử dụng cho cả các lớp giả và các phần tử giả trong CSS2 và CSS1.

Để tương thích ngược, cú pháp dấu hai chấm có thể chấp nhận được đối với các phần tử giả CSS2 và CSS1.

* The ::first-line Pseudo-element

Phần tử giả ::first-line được sử dụng để thêm một phong cách đặc biệt vào dòng đầu tiên của văn bản.

Ví dụ sau định dạng dòng đầu tiên của văn bản trong tất cả phần tử <p> :

p::first-line {  
  color: #ff0000;  
  font-variant: small-caps;  
}

Lưu ý : Các phần tử giả ::first-line chỉ có thể được áp dụng cho các phần tử ở mức khối.

Các thuộc tính sau áp dụng cho các phần tử giả ::first-line :

* Các thuộc tính font
* Thuộc tính color
* Thuộc tính background
* Word-spacing
* Letter-spacing
* Vertical-align
* Text-trànorm
* Line-height
* Clear
* The ::first-letter Pseudo-element

Các phần tử giả ::first-letter được sử dụng để thêm một phong cách đặc biệt để các chữ cái đầu tiên của một văn bản.

Ví dụ sau định dạng chữ cái đầu tiên của văn bản trong tất cả các phần tử <p> :

p::first-letter {  
  color: #ff0000;  
  font-size: xx-large;  
}

Lưu ý : Các phần tử giả ::first-letter chỉ có thể được áp dụng cho các yếu tố ở mức khối.

Các thuộc tính sau áp dụng đối với các phần tử ::first-letter :

* Thuộc tính font
* Thuộc tính color
* Thuộc tính background
* Thuộc tính margin
* Thuộc tính padding
* Thuộc tính border
* Text-decoration
* Vertical-align (chỉ khi “float” là “none”)
* Text-transform
* Line-height
* Float
* Clear
* Pseudo-elements và CSS Classes

Thành phần giả có thể được kết hợp với các lớp CSS.

p.intro::first-letter {  
  color: #ff0000;  
  font-size:200%;  
}

Ví dụ trên hiển thị các chữ cái đầu tiên của đoạn văn với class= “intro”, màu đỏ và trong một kích thước lớn hơn.

* Multiple Pseudo-elements

Một số thành phần giả cũng có thể được kết hợp.

Trong ví dụ sau, chữa cái đầu tiên của đoạn văn sẽ có màu đỏ, với kích thước phông chữ xx-large. Phần còn lại của dòng đầu tiên sẽ có màu xanh, và trong mũ nhỏ. Phần còn lại sẽ có màu và font chữ mặc định.

p::first-letter {  
  color: #ff0000;  
  font-size: xx-large;  
}  
  
p::first-line {  
  color: #0000ff;  
  font-variant: small-caps;  
}

* CSS – The ::before Pseudo-element

Phần tử giả::before có thể được sử dụng để chèn một số nội dung trước nội dung của một phần tử.

Ví dụ sau chèn một hình ảnh trước nội dung của mỗi phần tử <h1> :

h1::before {  
  content: url(smiley.gif);  
}

* CSS – The ::after Pseudo-element

Phần tử giả ::after có thể được sử dụng để chèn một số nội dung sau nội dung của một phần tử.

Ví dụ sau chèn một hình ảnh sau nội dung của phần tử <h1> :

h1::after {  
  content: url(smiley.gif);  
}

* CSS – The ::selection Pseudo-element

Phần tử giả ::selection so khớp với phần của phần tử được lựa chọn bởi người dùng.

Các thuộc tính CSS sau đây có thể được áp dụng cho phần tử giả ::selection : color, background, cusor và ouline.

Ví dụ sạu đây làm cho các văn bản đã chọn màu đỏ trên nền màu vàng:

::selection {  
  color: red;  
  background: yellow;  
}

***2.3.1.29 CSS Opacity***

Các thuộc tính opacity quy định các opacity/tính trong suốt của một phần tử.

* Transparent Image

Các thuộc tính opacity có thể mất một giá trị 0.0-1.0. Giá trị càng thấp, tính trong suốt càng cao:



Chú ý : IE8 và trước đó sử dụng filter:alpha(opacity=x). X có thể mất một giá trị từ 0-100. Giá trị nhỏ hơn khiến phần tử trở lên trong suốt hơn.

img {  
  opacity: 0.5;  
  filter: alpha(opacity=50); /\* For IE8 and earlier \*/  
}

* Transparent Hover Effect

Thuộc tính opacity thường được sử dụng với các bộ chon :hover để thay đổi độ mờ đục khi di chuột qua :

img {  
  opacity: 0.5;  
  filter: alpha(opacity=50); /\* For IE8 and earlier \*/  
}  
  
img:hover {  
  opacity: 1.0;  
  filter: alpha(opacity=100); /\* For IE8 and earlier \*/  
}

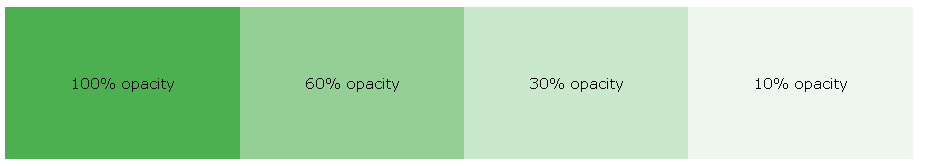
* Transparent box

Khi sử dụng thuộc tính opacity để thêm tính trong suốt cho nền tảng của một phần tử, tất cả các phần tử con của nó kế thừa tính trong suốt giống vậy. Điều này có thể làm cho văn bản bên trong một phần tử hoàn toàn trong suốt khó đọc:

div {  
  opacity: 0.3;  
  filter: alpha(opacity=30); /\* For IE8 and earlier \*/  
}

* Transparency using RGBA

Nếu bạn không muốn áp dụng opacity cho phần tử con, như trong ví dụ ỏe trên, sử dụng giá trị màu RGBA. Ví dụ sau áp dụng opacity cho màu nền và không áp dùng cho phần văn bản:



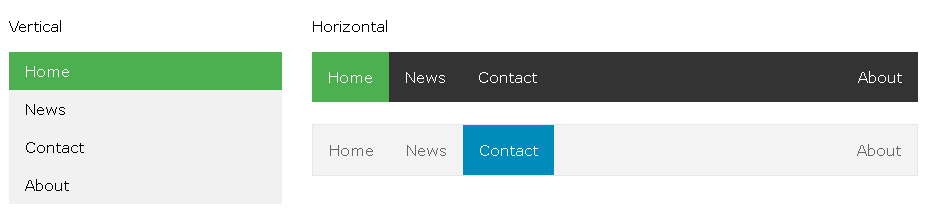
div {  
  background: rgba(76, 175, 80, 0.3) /\* Green background with 30% opacity \*/  
}

* Văn bản trong Transaprent box

<html>  
<head>  
<style>  
div.background {  
  background: url(klematis.jpg) repeat;  
  border: 2px solid black;  
}  
  
div.transbox {  
  margin: 30px;  
  background-color: #ffffff;  
  border: 1px solid black;  
  opacity: 0.6;  
  filter: alpha(opacity=60); /\* For IE8 and earlier \*/  
}  
  
div.transbox p {  
  margin: 5%;  
  font-weight: bold;  
  color: #000000;  
}  
</style>  
</head>  
<body>  
  
<div class="background">  
  <div class="transbox">  
    <p>This is some text that is placed in the transparent box.</p>  
  </div>  
</div>  
  
</body>  
</html>

***2.3.1.30 CSS Navigation Bar***

* Demo: navigation Bars



* Navigation Bars

Dễ dàng sử dụng điều hướng là điều quan trong đối với bất kỳ trang web nào.

Với CSS, bạn có thể chuyển đổi các menu nhàm chán thành các thanh điều hướng đẹp mắt.

* Navigation Bar = List of Links

Một thanh điều hướng cần chuẩn HTML làm nền tảng.

Trong ví dụ dưới đây, chúng ta sẽ xây dựng thanh điều hướng từ một danh sách HTML chuẩn.

Một thanh điều hướng cơ bản là một danh sách liên kết, vì vậy sử dụng phần tử <ul> và <li> khiến nó hoàn hảo hơn.

<ul>  
  <li><a href="default.asp">Home</a></li>  
  <li><a href="news.asp">News</a></li>  
  <li><a href="contact.asp">Contact</a></li>  
  <li><a href="about.asp">About</a></li>  
</ul>

Bây giờ chúng ta hãy loại bỏ các dấu chấm tròn và margin và padding từ danh sách:

ul {  
  list-style-type: none;  
  margin: 0;  
  padding: 0;  
}

* Vertical Navigation Bar

Để xây dựng một thanh điều hướng theo chiều dọc, bạn có thể tạo kiểu các phần tử <a> bên trong danh sách, ngoài đoạn mã trên:

li a {  
  display: block;  
  width: 60px;  
}

Bạn cũng có thể thiết lập độ rộng của <ul> và loại bỏ độ rộng của <a> vì chúng sẽ mất toàn bộ chiều rộng có sẵn khi được hiển thị như các phần tử khối. Điều này sẽ tạo ra kết quả tương tự như ví dụ trước:

ul {  
  list-style-type: none;  
  margin: 0;  
  padding: 0;  
  width: 60px;  
}  
  
li a {  
  display: block;  
}

* Ví dụ Vertical Navigation Bar

Tạo thanh điều hướng cơ bản với nền màu xám và thay đổi nền của các liên kết khi người dùng di chuột qua chúng :

ul {  
  list-style-type: none;  
  margin: 0;  
  padding: 0;  
  width: 200px;  
  background-color: #f1f1f1;  
}  
  
li a {  
  display: block;  
  color: #000;  
  padding: 8px 16px;  
  text-decoration: none;  
}  
  
/\* Change the link color on hover \*/  
li a:hover {  
  background-color: #555;  
  color: white;  
}

* Active/Current Navigation Link

Thêm một lớp “active” vào liên kết hiện tại để cho người dùng biết anh ấy/cô ấy đang ở trang nào:

.active {  
  background-color: #4CAF50;  
  color: white;  
}

* Center Links & Add Borders

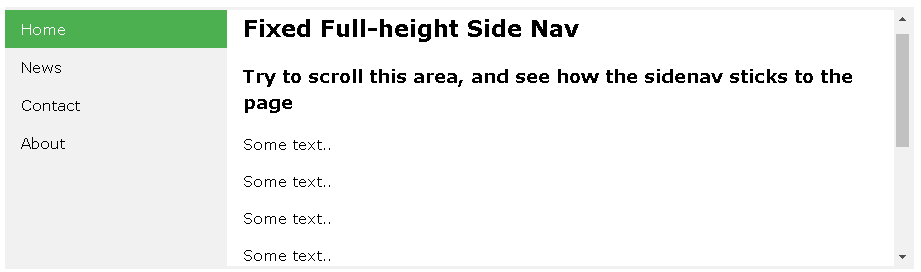
Thêm text-align:center vào thẻ <li> hoawck <a> để căn giữa các liên kết.

Thêm thuộc tính border vào <ul> thêm một đường viền xung quanh thanh điều hướng. Nếu bạn muốn thêm đường viền bên trong thanh điều hướng, hãy thêm border-bottom vào tất cả các phần tử <li>, ngoại trừ phần tử cuối cùng:

ul {  
  border: 1px solid #555;  
}  
  
li {  
  text-align: center;  
  border-bottom: 1px solid #555;  
}  
  
li:last-child {  
  border-bottom: none;  
}

* Full-height Fixed Vertical Navbar

Tạo một full-height, ngay bên cạnh thanh điều hướng:



ul {  
  list-style-type: none;  
  margin: 0;  
  padding: 0;  
  width: 25%;  
  background-color: #f1f1f1;  
  height: 100%; /\* Full height \*/  
  position: fixed; /\* Make it stick, even on scroll \*/  
  overflow: auto; /\* Enable scrolling if the sidenav has too much content \*/  
}

* Horizontal Navigation Bar

Có hai cách để tạo một thanh điều hướng ngang. Sử dụng danh sách *inline* hoặc *floating.*

* Danh sách Inline

Một cách để xây dựng một thanh điều hướng ngang là xác định phần tử <li> nội tuyến, ngoài các mã “chuẩn” bên trên:

li {  
  display: inline;  
}

* Danh sách Floating

Một cách khác để xây dựng một thanh điều hướng ngang là thả nổi phần tử <li>, và chỉ định bố cục cho liên kết điều hướng:

li {  
  float: left;  
}  
  
a {  
  display: block;  
  padding: 8px;  
  background-color: #dddddd;  
}

* Ví dụ Horizontal Navigation Bar

Tạo một thanh điều hướng ngang cơ bản với màu nền tối và thay đổi màu nền của các liên kết khi người dùng di chuột qua chúng:

ul {  
  list-style-type: none;  
  margin: 0;  
  padding: 0;  
  overflow: hidden;  
  background-color: #333;  
}  
  
li {  
  float: left;  
}  
  
li a {  
  display: block;  
  color: white;  
  text-align: center;  
  padding: 14px 16px;  
  text-decoration: none;  
}  
  
/\* Change the link color to #111 (black) on hover \*/  
li a:hover {  
  background-color: #111;  
}

* Active/Current Navigation Link
* Thêm một lớp “active” vào liên kết hiện tại để cho người dùng biết anh ấy/cô ấy đang ở trang nào:

.active {  
  background-color: #4CAF50;  
}

* Right-Align Links

Right-align links bằng cách nổi các mục danh sách sang bên phải (float:right; ):

<ul>  
  <li><a href="#home">Home</a></li>  
  <li><a href="#news">News</a></li>  
  <li><a href="#contact">Contact</a></li>  
  <li style="float:right"><a class="active" href="#about">About</a></li>  
</ul>

* Border Dividers

Thêm thuộc tính border-right vào <li> để tạo vách ngăn liên kết:

/\* Add a gray right border to all list items, except the last item (last-child) \*/  
li {  
  border-right: 1px solid #bbb;  
}  
  
li:last-child {  
  border-right: none;  
}

* Cố định thanh điều hướng

Đặt thanh điều hướng ở trên cùng hoặc dưới cùng của trang, ngay cả khi người dùng cuộn trang :

**Fixed Top**

ul {  
  position: fixed;  
  top: 0;  
  width: 100%;  
}

Fixed Buttom

ul {  
  position: fixed;  
  bottom: 0;  
  width: 100%;  
}

Chú ý: vị trí cố định có thể không hoạt động đúng trên các thiết bị di động.

* Gray Horizontal Navbar



ul {  
  border: 1px solid #e7e7e7;  
  background-color: #f3f3f3;  
}  
  
li a {  
  color: #666;  
}

* Sticky Navbar

Thêm position:sticky; vào <ul> để tạo một thanh điều hướng dính.

ul {  
  position: -webkit-sticky; /\* Safari \*/  
  position: sticky;  
  top: 0;  
}

Chú ý: Internet Explorer, Edge 15 và các phiên bản trước đó không hỗ trợ định vị dính, Safari đòi hỏi một tiền tố -webkit-. Bạn cũng phải chỉ định ít nhất một trong các vị trí top, right bottom hoặc left để định vị dính để làm việc.

***2.3.1.31 CSS Dropdowns***

Tạo một danh sách thả xuống hoverable với CSS.

* Basic Dropdown

Tạo một hộp thả xuống xuất hiện khi người dùng di chuột lên một phần tử.

<style>  
.dropdown {  
  position: relative;  
  display: inline-block;  
}  
  
.dropdown-content {  
  display: none;  
  position: absolute;  
  background-color: #f9f9f9;  
  min-width: 160px;  
  box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);  
  padding: 12px 16px;  
  z-index: 1;  
}  
  
.dropdown:hover .dropdown-content {  
  display: block;  
}  
</style>  
  
<div class="dropdown">  
  <span>Mouse over me</span>  
  <div class="dropdown-content">  
    <p>Hello World!</p>  
  </div>  
</div>

Giải thích:

**HTML)** sử dụng bất kỳ phần tử để mở nội dung thả xuống, ví dụ như một phần tử <span> hoặc <button>.

Sử dụng một phần tử Container( như <div>) để tạo ra nội dung thả xuống và thêm bất cứ điều gì bạn muốn bên trong nó.

Bọc một phần tử <div> xung quanh các phần tử để xác định vị trí các nội dung thả xuông một cách chính xác với CSS.

CSS) lớp .dropdown sử dụng position:relative, cần thiết khi chúng ta muốn các nội dung thả xuống được đặt ngay bên dưới nút thả xuống ( sử dụng position:absolute).

Lớp .dropdown-content giữ nội dung thả xuống thực tế. Nó được ẩn theo mặc định và sẽ được hiển thị khi di chuột. Lưu ý min-width được thiết lập để 160 px. Có thể tự do thay đổi min-width. Nếu bạn muốn chiều rộng của nội dung thả xuống rộng như vào nút thả xuống, thiết lập width=100% (và overflow:auto cho phép di chuyển trên màn hình nhỏ).

Thay vì sử dụng đường viền, chúng ta sử dụng thuộc tính box-shadow để làm cho menu thả xuống trông giống như một “thẻ”.

Selector :hover được sử dụng để hiển thị các trình đơn thả xuống khi người dùng di chuyển chuột qua nút thả xuống.

* Dropdown Menu

Tạo một menu thả xuống cho phép người dùng chọn một tùy chọn từ một danh sách:

<style>  
/\* Style The Dropdown Button \*/  
.dropbtn {  
  background-color: #4CAF50;  
  color: white;  
  padding: 16px;  
  font-size: 16px;  
  border: none;  
  cursor: pointer;  
}  
  
/\* The container <div> - needed to position the dropdown content \*/  
.dropdown {  
  position: relative;  
  display: inline-block;  
}  
  
/\* Dropdown Content (Hidden by Default) \*/  
.dropdown-content {  
  display: none;  
  position: absolute;  
  background-color: #f9f9f9;  
  min-width: 160px;  
  box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);  
  z-index: 1;  
}  
  
/\* Links inside the dropdown \*/  
.dropdown-content a {  
  color: black;  
  padding: 12px 16px;  
  text-decoration: none;  
  display: block;  
}  
  
/\* Change color of dropdown links on hover \*/  
.dropdown-content a:hover {background-color: #f1f1f1}  
  
/\* Show the dropdown menu on hover \*/  
.dropdown:hover .dropdown-content {  
  display: block;  
}  
  
/\* Change the background color of the dropdown button when the dropdown content is shown \*/  
.dropdown:hover .dropbtn {  
  background-color: #3e8e41;  
}  
</style>  
  
<div class="dropdown">  
  <button class="dropbtn">Dropdown</button>  
  <div class="dropdown-content">  
    <a href="#">Link 1</a>  
    <a href="#">Link 2</a>  
    <a href="#">Link 3</a>  
  </div>  
</div>

* Right-align Dropdown Content

Nếu bạn muốn menu thả xuống đi từ phải qua trái, thêm right:0;

.dropdown-content {  
  right: 0;  
}

***2.3.1.32 CSS Image Gallery***

CSS có thể được sử dụng để tạo ra một thư viện hình ảnh.

* Thư viện hình ảnh

Các bộ sưu tập hình ảnh sau đây được tạo ra với CSS:

<html>  
<head>  
<style>  
div.gallery {  
  margin: 5px;  
  border: 1px solid #ccc;  
  float: left;  
  width: 180px;  
}  
  
div.gallery:hover {  
  border: 1px solid #777;  
}  
  
div.gallery img {  
  width: 100%;  
  height: auto;  
}  
  
div.desc {  
  padding: 15px;  
  text-align: center;  
}  
</style>  
</head>  
<body>  
  
<div class="gallery">  
  <a target="\_blank" href="img\_5terre.jpg">  
    <img src="img\_5terre.jpg" alt="Cinque Terre" width="600" height="400">  
  </a>  
  <div class="desc">Add a description of the image here</div>  
</div>  
  
<div class="gallery">  
  <a target="\_blank" href="img\_forest.jpg">  
    <img src="img\_forest.jpg" alt="Forest" width="600" height="400">  
  </a>  
  <div class="desc">Add a description of the image here</div>  
</div>  
  
<div class="gallery">  
  <a target="\_blank" href="img\_lights.jpg">  
    <img src="img\_lights.jpg" alt="Northern Lights" width="600" height="400">  
  </a>  
  <div class="desc">Add a description of the image here</div>  
</div>  
  
<div class="gallery">  
  <a target="\_blank" href="img\_mountains.jpg">  
    <img src="img\_mountains.jpg" alt="Mountains" width="600" height="400">  
  </a>  
  <div class="desc">Add a description of the image here</div>  
</div>  
  
</body>  
</html>

***2.3.1.33CSS Image Sprites***

* Image Sprites

Một sprite hình ảnh là một bộ sưu tập các hình ảnh được đưa vào một hình ảnh duy nhất.

Một trang web với nhiều hình ảnh có thể mất nhiều thời gian để tải và tạo ra nhiều yêu cầu máy chủ.

Sử dụng sprites hình ảnh sẽ làm giảm số lượng yêu cầu máy chủ và tiết kiệm băng thông.

* Ví dụ

Thay vì sử dụng 3 hình ảnh riêng biệt, chúng ta sử dụng một hình ảnh duy nhất (“img\_navsprites.gif):



Với CSS, chúng ta có thể chỉ hiển thị một phần của hình ảnh chúng ta cần.

Trong ví dụ dưới đây, CSS xác định cụ thể một phần của hình ảnh “img\_navsprites.gif” được hiển thị:

#home {  
  width: 46px;  
  height: 44px;  
  background: url(img\_navsprites.gif) 0 0;  
}

* Image Sprites – Create a Navigation List

Chúng ta muốn sử dụng hình ảnh (“img\_navsprites.gif”) để tạo ra một danh sách điều hướng. Chúng ta sẽ sử dụng một danh sách HTML, bởi vì nó có thể là một liên kết và cũng hỗ trợ hình nền:

#navlist {  
  position: relative;  
}  
  
#navlist li {  
  margin: 0;  
  padding: 0;  
  list-style: none;  
  position: absolute;  
  top: 0;  
}  
  
#navlist li, #navlist a {  
  height: 44px;  
  display: block;  
}  
  
#home {  
  left: 0px;  
  width: 46px;  
  background: url('img\_navsprites.gif') 0 0;  
}  
  
#prev {  
  left: 63px;  
  width: 43px;  
  background: url('img\_navsprites.gif') -47px 0;  
}  
  
#next {  
  left: 129px;  
  width: 43px;  
  background: url('img\_navsprites.gif') -91px 0;  
}

* Sprites Image – Hover Effect

Bây giờ chúng ta muốn thêm một hiệu ứng di chuột vào danh sách chuyển hướng của mình.

Các bộ chọn :hover có thể được sử dụng trên tất cả các yếu tố, không chỉ trên các liên kết.

Hình ảnh mới của chúng ta (“img\_navsprites\_hover.gif”) chứa ba hình ảnh điều hướng và ba hình ảnh để sự dụng cho các hiệu ứng di chuột.



Bởi vì đây là một hình ảnh duy nhất, và không giống với 6 tập tin riêng biệt, sẽ có sự chậm trễ không tải khi di chuột qua hình ảnh.

Chúng ta chỉ thêm 3 dòng code để thêm các hiệu ứng di chuột:

#home a:hover {  
  background: url('img\_navsprites\_hover.gif') 0 -45px;  
}  
  
#prev a:hover {  
  background: url('img\_navsprites\_hover.gif') -47px -45px;  
}  
  
#next a:hover {  
  background: url('img\_navsprites\_hover.gif') -91px -45px;  
}

***2.3.1.34 CSS Attr Selectors***

* CSS [attribute] Selector

Các [attribute] selector được sử dụng để chọn các yếu tố với một thuộc tính cụ thể.

Ví dụ sau đây chọn tất cả các phần tử <a> với thuộc tính target :

a[target] {  
  background-color: yellow;  
}

* CSS [attribute= “value”] selector

[attribute= “value”] selector được sử dụng để chọn các phần tử với một thuộc tính và giá trị quy định.

Ví dụ sau sẽ chọn tất cả các phần tử <a> với thuộc tính target = “\_blank” :

a[target="\_blank"] {  
  background-color: yellow;  
}

* CSS [attribute~= “value”] selector

[attribute~= “value”] selector được sử dụng để chọn các phần tử với một giá trị thuộc tính có chứa một từ nhất định.

Ví dụ sau sẽ chọn tất cả các phần tử với một thuộc tính tiêu đề có chứa một danh sách không gian tách từ, một trong số đó là “flower” :

[title~="flower"] {  
  border: 5px solid yellow;  
}

* CSS [attribute|= “value”] selector

[attribute|= “value”] selector được sử dụng để chọn các phần tử với các thuộc tính quy định với các giá trị quy định.

Ví dụ sau sẽ chọn tất carcacs phần tử với nột giá trị thuộc tính lớp bắt đầu với “top” :

Lưu ý: giá trị phải là toàn bộ một từ, hoặc một mình, giống như class=”top”, hoặc theo sau là một dấu gạch ngang (-), giống như class= “top-text”

[class|="top"] {  
  background: yellow;  
}

* CSS [attribute^= “value”] selector

[attribute^= “value”] selector được sử dụng để chọn các phần tử có giá trị thuộc tính bắt đầu với một giá trị xác định.

Ví dụ sau sẽ chọn tất cả các phần tử với một giá trị thuộc tính lớp bắt đầu với “top” :

[class^="top"] {  
  background: yellow;  
}

* CSS [attribute $= “value”] selector

[attribute $= “value”] selector được sử dụng để chọn các phần tử có giá trị thuộc tính kết thúc bằng một giá trị xác định.

Ví dụ sau sẽ chọn tất cả các phần tử với một giá trị thuộc tính lớp kết thúc bằng “test” :

[class$="test"] {  
  background: yellow;  
}

* CSS [attribute \*= “value”] selector

[attribute \*= “value”] selector được sử dụng để chọn các phần tử có giá trị thuộc tính chứa một giá trị xác định.

Ví dụ sau sẽ chọn tất cả các yếu tố với một gái trị thuộc tính lớp có chưa “te” :

[class\*="te"] {  
  background: yellow;  
}

* Styling Forms

Các bộ chọn thuộc tính có thể hữu ích cho các hình thức định dạng mà không cần class hoặc id :

input[type="text"] {  
  width: 150px;  
  display: block;  
  margin-bottom: 10px;  
  background-color: yellow;  
}  
  
input[type="button"] {  
  width: 120px;  
  margin-left: 35px;  
  display: block;  
}

***2.3.1.35 CSS Forms***

Diện mạo của một form HTML có thể được cải thiện rất nhiều với CSS.



* Styling Input Fields

Sử dụng thuộc tính width để xác định độ rộng của trường input.

input {  
  width: 100%;  
}

Ví dụ trên áp dụng cho tất cả các phần tử <input>. Nếu bạn chỉ muốn định dạng một kiểu input xác định, bạn có thể sử dụng bộ chọn các thuộc tính:

* input[type=text] - chỉ chọn trường văn bản.
* input[type=password] - chỉ chọn trường mật khẩu.
* input[type=number] - chỉ chọn trường số.
* Padded Inputs

Sử dụng thuộc tính padding để thêm không gian bên trong trường văn bản.

Khi bạn có nhiều đầu vào phía sau, bạn cũng có thể muốn thêm một số lề, để thêm không gian bên ngoài chúng:

input[type=text] {  
  width: 100%;  
  padding: 12px 20px;  
  margin: 8px 0;  
  box-sizing: border-box;  
}

Chú ý rằng chúng ta đã cài đặt thuộc tính box-sizing cho border-box. Điều này đảm bảo rằng phần đệm và đường viền cuối cùng được bao trong tổng chiều rộng và chiều cao của các phần tử.

* Bordered inputs

Sử dụng thuộc tính border để thay đổi kích thước đường viền và màu sắc, sử dụng thuộc tính border-radius để thêm góc tròn.

input[type=text] {  
  border: 2px solid red;  
  border-radius: 4px;  
}

Nếu bạn chỉ muốn có một đường viền phía dưới , sử dụng thuộc tính border-bottom:

input[type=text] {  
  border: none;  
  border-bottom: 2px solid red;  
}

* Colored Inputs

Sử dụng thuộc tính background-color để thêm màu nền của trường đầu vào, và thuộc tính color để thay đổi màu sắc văn bản:

input[type=text] {  
  background-color: #3CBC8D;  
  color: white;  
}

* Focus Inputs

Theo mặc định, một số trình duyệt sẽ thêm một đường viền màu xanh xung quanh phần tử input khi nó được focus (nhấp vào). Bạn có thể loại bỏ hành vi này bằng cách thêm outline:none; vào input.

Sử dụng các :focus selector để làm một điều gì đó với trường đầu vào khi nó được focus:

input[type=text]:focus {  
  background-color: lightblue;  
}

input[type=text]:focus {  
  border: 3px solid #555;  
}

* Input with icon/image

Nếu bạn muốn có một biểu tượng bên trong input, sử dụng thuộc tính background-image và đặt nó với thuộc tính background-position. Lưu ý rằng chúng ta thêm một phần đệm lớn bên trái để dành chỗ cho biểu tượng:

input[type=text] {  
  background-color: white;  
  background-image: url('searchicon.png');  
  background-position: 10px 10px;  
  background-repeat: no-repeat;  
  padding-left: 40px;  
}

* Animated Search Input

Trong ví dụ này chúng ta sử dụng thuộc tính CSS transition để animate chiều rộng của đầu vào tìm kiếm khi nó được focus.

input[type=text] {  
  -webkit-transition: width 0.4s ease-in-out;  
  transition: width 0.4s ease-in-out;  
}  
  
input[type=text]:focus {  
  width: 100%;  
}

* Styling textareas

Sử dụng các thuộc tính resize để ngăn chặn textareas khỏi bị thay đổi kích thước :

textarea {  
  width: 100%;  
  height: 150px;  
  padding: 12px 20px;  
  box-sizing: border-box;  
  border: 2px solid #ccc;  
  border-radius: 4px;  
  background-color: #f8f8f8;  
  resize: none;  
}

* Styling Select Menu

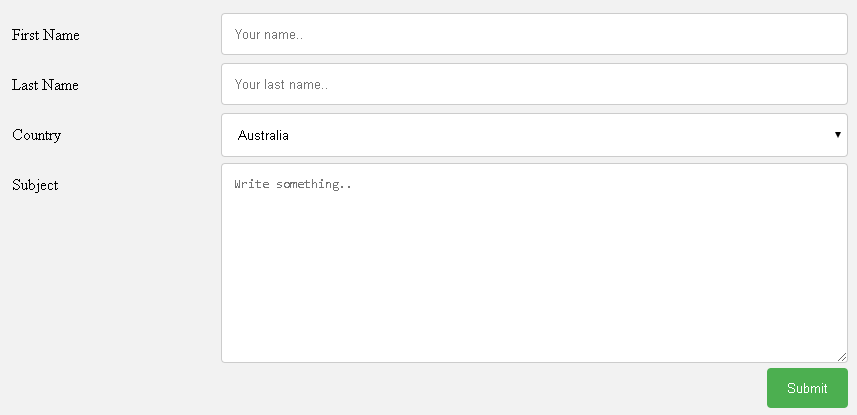
select {  
  width: 100%;  
  padding: 16px 20px;  
  border: none;  
  border-radius: 4px;  
  background-color: #f1f1f1;  
}

* Styling Input Buttons

input[type=button], input[type=submit], input[type=reset] {  
  background-color: #4CAF50;  
  border: none;  
  color: white;  
  padding: 16px 32px;  
  text-decoration: none;  
  margin: 4px 2px;  
  cursor: pointer;  
}  
  
/\* Tip: use width: 100% for full-width buttons \*/

* Responsive Form

Thay đổi kích thước cửa sổ trình duyệt để xem hiệu quả. Khi màn hình có độ rộng nhỏ hơn 600px, làm cho hai cột xếp chồng lên nhau thay vì cạnh nhau.



***2.3.1.36 CSS Counters***

CSS counters là biến duy trì bởi CSS có giá trị có thể được tăng lên bởi các quy tắc CSS (để theo dõi bao nhiêu lần chúng được sử dụng). Bộ đếm cho phép bạn điều chỉnh sự xuất hiện của nội dung dựa trên vị trí của nó trong tài liệu.

* Tự động đánh số với Counters

CSS Counters cũng giống như “biến”. Các giá trị biến có thể được tăng lên bởi các quy tắc CSS (mà sẽ theo dõi bao nhiêu lần chúng được sử dụng).

Để làm việc với CSS counters, chúng ta sẽ sử dụng các thuộc tính sau:

* Counter-reset – tạo ra hoặc reset bộ đếm
* Counter-increment – tăng giá trị truy cập
* Content – chèn một nội dung.
* Counter() hoặc hàm counters() – thêm giá trị của bộ đếm vào một phần tử.

Để sử dụng CSS counter, trước tiên nó phải được tạo với counter-reset.

Ví dụ dưới đây tạo một bộ đếm cho một trang, sau đó tăng giá trị truy cập cho mỗi phần tử <h2> và thêm “Section <value of the counter>:” vào đầu mỗi phần tử <h2> :

body {  
  counter-reset: section;  
}  
  
h2::before {  
  counter-increment: section;  
  content: "Section " counter(section) ": ";  
}

* Nesting Counters

Ví dụ sau đây tạo một bộ đếm cho trang (section) và một bộ đếm cho mỗi phần tử <h1> (subsection) . Bộ đếm “section” sẽ được đếm cho từng phần tử <h1> với “Section <value of the section counter>” và bộ đếm “subsection” sẽ được đếm cho mỗi phần tử <h2> với “<value of the section counter>.<value of the subsection counter>” .

body {  
  counter-reset: section;  
}  
  
h1 {  
  counter-reset: subsection;  
}  
  
h1::before {  
  counter-increment: section;  
  content: "Section " counter(section) ". ";  
}  
  
h2::before {  
  counter-increment: subsection;  
  content: counter(section) "." counter(subsection) " ";  
}

Một bộ đếm cũng có thể hữu ích để tạo danh sách được phác thảo vì một phiên bản mới của bộ đếm được tạo ra tự động trong các phần tử con. Ở đây chúng ta sử dụng hàm counters() để chèn một chuỗi giữa các mức khác nhau của các bộ đếm lồng nhau:

ol {  
  counter-reset: section;  
  list-style-type: none;  
}  
  
li::before {  
  counter-increment: section;  
  content: counters(section,".") " ";  
}

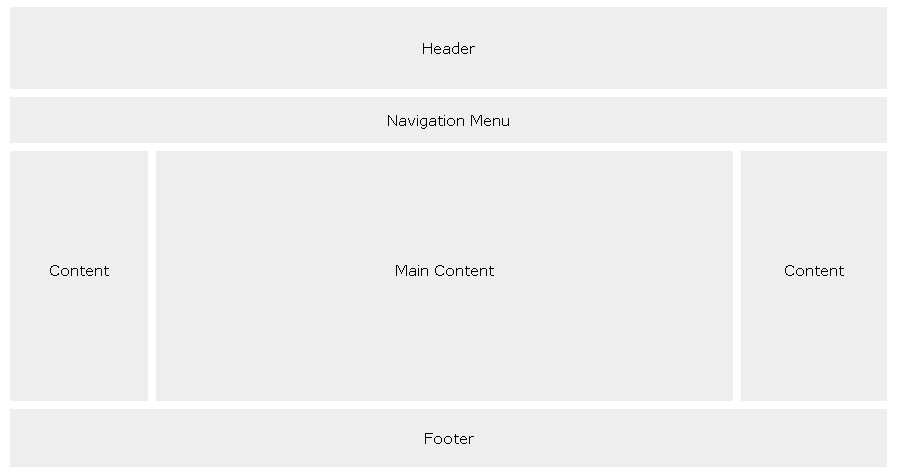
* Các thuộc tính của CSS Counter

|  |  |
| --- | --- |
| **Thuộc tính** | **Mô tả** |
| [content](https://www.w3schools.com/cssref/pr_gen_content.asp) | Sử dụng với phần tử giả ::before và ::after, để chèn nội dung được tạo. |
| [counter-increment](https://www.w3schools.com/cssref/pr_gen_counter-increment.asp) | Tăng một hoặc nhiều giá trị truy cập. |
| [counter-reset](https://www.w3schools.com/cssref/pr_gen_counter-reset.asp) | Tạo hoặc đặt lại một hay nhiều bộ đếm/ |

***2.3.1.37 CSS Website Layout***

* + Website Layout

Một trang web thường được chia thành các phần header, menu, content và footer.



Có rất nhiều các bộ cục thiết kế khác nhau để lựa chọn. tuy nhiên, bố cục trên là một trong những bố cục phổ biến nhất.

* + Header

Một tiêu đề thường nằm ở phía trên cùng của trang web (hoặc ngay phía dưới đỉnh một menu điều hướng). Nó thường chứa một logo hoặc tên website:

.header {  
  background-color: #F1F1F1;  
  text-align: center;  
  padding: 20px;  
}

* + Thanh điều hướng

Một thanh điều hướng chứa một danh sách các liên kết để giúp người dùng điều hướng thông qua trang web của bạn:

/\* The navbar container \*/  
.topnav {  
  overflow: hidden;  
  background-color: #333;  
}  
  
/\* Navbar links \*/  
.topnav a {  
  float: left;  
  display: block;  
  color: #f2f2f2;  
  text-align: center;  
  padding: 14px 16px;  
  text-decoration: none;  
}  
  
/\* Links - change color on hover \*/  
.topnav a:hover {  
  background-color: #ddd;  
  color: black;  
}

* + Content

Cách bố trí trong phần này thường phụ thuộc vào mục đích của người dùng. Các bố trí phổ biến nhất là một (hoặc kết hợp chúng) trong các cách sau:

* 1-column ( thường dùng cho trình duyệt di động)
* 2-column ( thường sử dụng cho máy tính bảng và máy tính xách tay)
* 3-column (chỉ được sử dụng cho máy tính để bàn)



Chúng ta sẽ tạo ra một bố cục 3-column, và thay đổi nó thành bố cục 1-column trên màn hình nhỏ hơn:

/\* Create three equal columns that floats next to each other \*/  
.column {  
  float: left;  
  width: 33.33%;  
}  
  
/\* Clear floats after the columns \*/  
.row:after {  
  content: "";  
  display: table;  
  clear: both;  
}  
  
/\* Responsive layout - makes the three columns stack on top of each other instead of next to each other on smaller screens (600px wide or less) \*/  
@media screen and (max-width: 600px) {  
  .column {  
    width: 100%;  
  }  
}

Chú ý: để tạo một bố cục 2-column, thay đổi chiều rộng là 50%. Để tạo một bố cục 4-column, sử dụng 25%...

* + Unequal Columns

Nội dung chính là phần lớn nhất và quan trọng nhất của trang web của bạn.

Nó thường phổ biến với chiều rộng cột không bằng nhau, do đó hầu hết không gian được dành cho nội dung chính. Nội dung bên cạnh (nếu có) thường được sử dụng làm điều hướng thay thế hoặc để chỉ định thông tin liên quan đến nội dung chính. Thay đổi độ rộng theo ý muốn, nhưng nên nhớ rằng tổng các độ rộng tối đa 100% :

.column {  
  float: left;  
}  
  
/\* Left and right column \*/  
.column.side {  
  width: 25%;  
}  
  
/\* Middle column \*/  
.column.middle {  
  width: 50%;  
}  
  
/\* Responsive layout - makes the three columns stack on top of each other instead of next to each other \*/  
@media screen and (max-width: 600px) {  
  .column.side, .column.middle {  
    width: 100%;  
  }  
}

* + Footer

Footer được đặt dưới cuối trang web của bạn. Nó thường chứa thông tin như bản quyền và thông tin liên hệ :

.footer {  
  background-color: #F1F1F1;  
  text-align: center;  
  padding: 10px;  
}

* + Responsive Website Layout

Bằng cách sử dụng các mã CSS ở trên, chúng ta đã tạo ra một bố cục trang web đáp ứng, khác nhau giữa 2 cột và chiều rộng toàn cột tùy thuộc vào chiều rộng của màn hình.

***2.3.1.38 CSS Units***

* + CSS units

CSS có các đơn vị khác nhau để thể hiện chiều dài.

Nhiều thuộc tính CSS có giá trị “chiều dài”, như width, margin, padding, font-size,..

Chiều dài là một con số theo sau là một đơn vị chiều dài như 10px, 2em…

Một khoảng trắng không thể xuất hiện giữa số lượng và đơn vị. Tuy nhiên, nếu giá trị là 0, đơn vị có thể được bỏ qua.

Đối với một số thuộc tính CSS, độ dài âm được cho phép.

Có hai loại đơn vị chiều dài : tuyệt đối và tương đối.

* + Chiều dài tuyệt đối

Các đơn vị chiều dài tuyệt đối được cố định và chiều dài được thể hiện trong bất kỳ số nào sẽ xuất hiện chính xác với kích thước đó.

Đơn vị độ dài tuyệt đối không được khuyến nghị sử dụng trên màn hình, vì kích thước màn hình rất khác nhau. Tuy nhiên chúng có thể dược sử dụng nếu phương tiện đầu ra được biết đến, chẳng hạn như đối với bố cục in.

|  |  |
| --- | --- |
| **Đơn vị** | **Mô tả** |
| cm | centimeters |
| mm | millimeters |
| in | inches (1in = 96px = 2.54cm) |
| px \* | pixels (1px = 1/96th of 1in) |
| pt | points (1pt = 1/72 of 1in) |
| pc | picas (1pc = 12 pt) |

* + Chiều dài tương đối

Đơn vị độ dài tương đối xác định độ dài tương đối với thuộc tính độ dài khác. Đơn vị độ dài tương đối tỷ lệ tốt hơn giữa các phương tiện kết xuất khác nhau.

|  |  |
| --- | --- |
| **Đơn vị** | **Mô tả** |
| em | Liên quan đến kích thước phông chữ của phần tử ( 2m có nghĩa là gấp 2 lần kích thước của phông chữ hiện tại) |
| ex | Liên quan đến chiều cao x của phông chữ hiện tại (ít khi sử dụng) |
| ch | Liên quan đến chiều rộng của “0”(zero) |
| rem | Liên quan đến kích thước phông chữ của phần tử gốc |
| vw | Liên quan đến 1% chiều rộng của khung nhìn. |
| vh | Liên quan đến 1% chiều cao của khung nhìn. |
| vmin | Liên quan đến 1% kích thước nhỏ hơn của khung nhìn |
| vmax | Liên quan đến 1% kích thước lớn hơn của khung nhìn |
| % | Liên quan đến phần tử cha |

* + Trình duyệt hỗ trợ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Length Unit | Chrome | Internet Explorer | Firefox | Safari | Opera |
| em, ex, %, px, cm, mm, in, pt, pc | 1.0 | 3.0 | 1.0 | 1.0 | 3.5 |
| ch | 27.0 | 9.0 | 1.0 | 7.0 | 20.0 |
| rem | 4.0 | 9.0 | 3.6 | 4.1 | 11.6 |
| vh, vw | 20.0 | 9.0 | 19.0 | 6.0 | 20.0 |
| vmin | 20.0 | 9.0\* | 19.0 | 6.0 | 20.0 |
| vmax | 26.0 | Not supported | 19.0 | 7.0 | 20.0 |

***2.3.1.39 CSS Specificity***

* + CSS Specificity là gì?

Nếu có hai hoặc nhiều xung đột về quy tắc CSS trỏ đến cùng một phần tử, trình duyệt sẽ tuân theo một số quy tắc để xác định quy tắc nào cụ thể nhất và nó sẽ thắng.

Hãy nghĩ về tính đặc hiệu như một điểm/ thứ hạng xác định khai báo kiểu nào cuối cùng được áp dụng cho một phần tử.

Selector phổ cập (\*) có độ đặc hiệu thấp trong khi ID selector có độ đặc hiệu cao.

* + Specificity Hierarchy

Mỗi selector có vị trí của nó trong hệ thống phân cấp cụ thể. Có 4 loại xác định mức độ đặc hiệu của một selector:

* Inline styles – một kiểu nội tuyến được gắn trực tiếp vào phần tử được tạo kiểu. Ví dụ: <h1 style= “color :#ffffff;”>.
* ID – id là một mã định danh duy nhất cho các phần tử trang như #navbar.
* Classes, attributes và pseudo-classes – danh mục này bao gồm .classes, [attributes] và pseudo-classes như :hover, :focus…
* Elements và pseudo-elements – danh mục này bao gồm tên phần tử và phần tử giả, ví dụ như h1, div, :before và :after
  + Làm thế nào để tính toán đặc hiệu?

Bắt đầu ở 0, thêm 1000 cho thuộc tính style, thêm 100 cho mỗi ID, thêm 10 cho mỗi thuộc tính, class hoặc pseudo-class, thêm 1 cho mỗi tên phần tử hoặc phần tử giả.

Xem xét ba đoạn mã sau:

A: h1  
B: #content h1  
C: <div id="content"><h1 style="color: #ffffff">Heading</h1></div>

Tính đặc hiệu của A là 1 (một phần tử)

Tính đặc hiệu của B là 101 ( một tham chiếu ID và một phần tử)

Tính đặc hiệu của C là 1000 (định dạng nội tuyến)

Từ 1<101<1000, quy tắc thứ 3 C có mức độ đặc hiệu cao hơn, do đó sẽ được áp dụng.

* + Specificity Rules

*Equal specificity: the latest rule counts* – nếu cùng một quy tắc được viết hai lần vào style sheet bên ngoài, thì quy tắc thấp hơn trong style sheet sẽ gần với phần tử được tạo kiểu hơn và do đó sẽ được áp dụng :

h1 {background-color: yellow;}  
h1 {background-color: red;}

Quy tắc sau sẽ luôn luôn được áp dụng.

*Selectors ID có tính đặc hiệu cao hơn các selector thuộc tính* – nhìn vào ví dụ sau:

div#a {background-color: green;}  
#a {background-color: yellow;}  
div[id=a] {background-color: blue;}

Quy tắc đầu tiên cụ thể hơn 2 quy tắc còn lại, do đó sẽ được áp dụng.

*Selector theo ngữ cảnh cụ thể hơn phần tử selector đơn* – style sheet kiểu nhúng gần với phần tử được tạo kiểu hơn. Vì vậy trong trường hợp sau đây quy tắc sau sẽ được áp dụng:

From external CSS file:  
#content h1 {background-color: red;}  
  
In HTML file:  
<style>  
#content h1 {  
  background-color: yellow;  
}  
</style>

*Selector class vượt mọi phần tử selector* – một selector class như .intro vượt h1, p, div….

.intro {background-color: yellow;}  
h1 {background-color: red;}

**2.3.2 CSS Advanced**

***2.3.2.1 CSS Rounded Corners***

*Góc làm tròn CSS*

Với thuộc tính bán kính đường viền CSS, bạn có thể cung cấp cho bất kỳ phần tử "góc tròn" nào.

Thuộc tính bán kính biên CSS

Thuộc tính bán kính đường viền CSS xác định bán kính của các góc của phần tử.

Mẹo: Thuộc tính này cho phép bạn thêm các góc tròn cho các phần tử!

Dưới đây là ba ví dụ:

1. Các góc được làm tròn cho một phần tử có màu nền được chỉ định:

2. Các góc được làm tròn cho một phần tử có viền:

3. Các góc được làm tròn cho một thành phần có hình nền:

Đây là mã:

#rcorners1 {  
  border-radius: 25px;  
  background: #73AD21;  
  padding: 20px;  
  width: 200px;  
  height: 150px;  
}  
#rcorners2 {  
  border-radius: 25px;  
  border: 2px solid #73AD21;  
  padding: 20px;  
  width: 200px;  
  height: 150px;  
}  
#rcorners3 {  
  border-radius: 25px;  
  background: url(paper.gif);  
  background-position: left top;  
  background-repeat: repeat;  
  padding: 20px;  
  width: 200px;  
  height: 150px;  
}

*Bán kính đường viền CSS - Chỉ định từng góc*

Bốn giá trị - Border-radius: 15px 50px 30px 5px; (giá trị đầu tiên áp dụng cho góc trên cùng bên trái, giá trị thứ hai áp dụng cho góc trên bên phải, giá trị thứ ba áp dụng cho góc dưới bên phải và giá trị thứ tư áp dụng cho góc dưới bên trái):

Ba giá trị - Border-radius: 15px 50px 30px; (giá trị đầu tiên áp dụng cho góc trên cùng bên trái, giá trị thứ hai áp dụng cho góc trên bên phải và dưới cùng bên trái và giá trị thứ ba áp dụng cho góc dưới bên phải):

Hai giá trị - Border-radius: 15px 50px; (giá trị đầu tiên áp dụng cho các góc trên cùng bên trái và dưới cùng bên phải và giá trị thứ hai áp dụng cho các góc trên cùng bên phải và dưới cùng bên trái):

Một giá trị - đường viền-bán kính: 15px; (giá trị áp dụng cho cả bốn góc, được làm tròn bằng nhau:

#rcorners1 {  
  border-radius: 15px 50px 30px 5px;  
  background: #73AD21;  
  padding: 20px;  
  width: 200px;  
  height: 150px;  
}  
#rcorners2 {  
  border-radius: 15px 50px 30px;  
  background: #73AD21;  
  padding: 20px;  
  width: 200px;  
  height: 150px;  
}  
#rcorners3 {  
  border-radius: 15px 50px;  
  background: #73AD21;  
  padding: 20px;  
  width: 200px;  
  height: 150px;  
}  
#rcorners4 {  
  border-radius: 15px;  
  background: #73AD21;  
  padding: 20px;  
  width: 200px;  
  height: 150px;  
}

*Thuộc tính góc tròn*



***2.3.2.2 CSS Border images***

*1*. Thuộc tính hình ảnh viền CSS

border-image Thuộc tính CSS cho phép bạn chỉ định một hình ảnh sẽ được sử dụng thay vì đường viền thông thường xung quanh một phần tử.

Tài sản có ba phần:

Hình ảnh được sử dụng làm đường viền

Nơi để cắt hình ảnh

Xác định xem các phần giữa nên được lặp đi lặp lại hoặc kéo dài

Chúng tôi sẽ sử dụng hình ảnh sau (được gọi là "Border.png"):

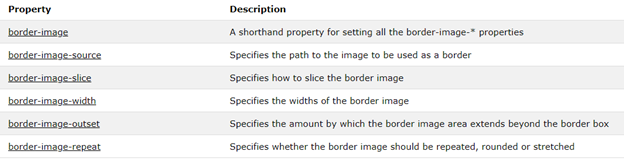
Các border-imagebất động sản có hình ảnh và lát nó thành chín phần, giống như một hội đồng quản trị tic-tac-toe. Sau đó, nó đặt các góc ở các góc, và các phần giữa được lặp lại hoặc kéo dài như bạn chỉ định.

2. CSS viền hình ảnh - Các giá trị lát khác nhau

Các giá trị lát khác nhau thay đổi hoàn toàn giao diện của đường viền

#borderimg1 {  
  border: 10px solid transparent;  
  padding: 15px;  
  border-image: url(border.png) 50 round;  
}  
#borderimg2 {  
  border: 10px solid transparent;  
  padding: 15px;  
  border-image: url(border.png) 20% round;  
}  
#borderimg3 {  
  border: 10px solid transparent;  
  padding: 15px;  
  border-image: url(border.png) 30% round;  
}

3. Thuộc tính hình ảnh viền CSS

**

***2.3.2.3 CSS Colors***

*1.* CSS nhiều nền

CSS cho phép bạn thêm nhiều ảnh nền cho một thành phần, thông qua thuộc background-imagetính.

Các hình nền khác nhau được phân tách bằng dấu phẩy và các hình ảnh được xếp chồng lên nhau, trong đó hình ảnh đầu tiên gần nhất với người xem.

Ví dụ sau có hai hình nền, hình ảnh đầu tiên là một bông hoa (được căn chỉnh ở phía dưới và bên phải) và hình ảnh thứ hai là một nền giấy (được căn chỉnh ở góc trên cùng bên trái):

#example1 {  
  background-image: url(img\_flwr.gif), url(paper.gif);  
  background-position: right bottom, left top;  
  background-repeat: no-repeat, repeat;  
}

Nhiều hình ảnh nền có thể được chỉ định bằng cách sử dụng các thuộc tính nền riêng lẻ (như trên) hoặc thuộc tính tốc backgroundký.

Ví dụ sau sử dụng thuộc tính tốc backgroundký (kết quả tương tự như ví dụ trên):

#example1 {  
  background: url(img\_flwr.gif) right bottom no-repeat, url(paper.gif) left top repeat;  
}

2. Kích thước nền CSS

background-sizeThuộc tính CSS cho phép bạn chỉ định kích thước của hình nền.

Kích thước có thể được chỉ định theo độ dài, tỷ lệ phần trăm hoặc bằng cách sử dụng một trong hai từ khóa: chứa hoặc che.

Ví dụ sau thay đổi kích thước hình nền thành nhỏ hơn nhiều so với hình ảnh gốc (sử dụng pixel):

#div1 {  
  background: url(img\_flower.jpg);  
  background-size: 100px 80px;  
  background-repeat: no-repeat;  
}

Hai giá trị có thể khác cho background-sizelà contain và cover.

Các contain từ khóa quy mô hình nền là càng lớn càng tốt (nhưng cả hai chiều rộng của nó và chiều cao của nó phải phù hợp với bên trong khu vực nội dung). Như vậy, tùy thuộc vào tỷ lệ của hình nền và khu vực định vị nền, có thể có một số khu vực nền không được bao phủ bởi hình nền.

Các cover từ khóa quy mô hình nền sao cho diện tích nội dung là hoàn toàn bao phủ bởi các hình nền (cả chiều rộng và chiều cao của nó bằng hoặc vượt quá diện tích nội dung). Như vậy, một số phần của hình nền có thể không hiển thị trong khu vực định vị nền.

Ví dụ sau minh họa việc sử dụng containvà cover:

#div1 {  
  background: url(img\_flower.jpg);  
  background-size: contain;  
  background-repeat: no-repeat;  
}  
#div2 {  
  background: url(img\_flower.jpg);  
  background-size: cover;  
  background-repeat: no-repeat;  
}

3.Xác định kích thước của nhiều ảnh nền

Các background-sizebất động sản cũng chấp nhận nhiều giá trị cho kích thước nền (sử dụng một danh sách bằng dấu phẩy), khi làm việc với nhiều nền.

Ví dụ sau có ba hình nền được chỉ định, với giá trị kích thước nền khác nhau cho mỗi hình ảnh:

#example1 {  
  background: url(img\_tree.gif) left top no-repeat, url(img\_flwr.gif) right bottom no-repeat, url(paper.gif) left top repeat;  
  background-size: 50px, 130px, auto;  
}

Các background-sizebất động sản cũng chấp nhận nhiều giá trị cho kích thước nền (sử dụng một danh sách bằng dấu phẩy), khi làm việc với nhiều nền.

Ví dụ sau có ba hình nền được chỉ định, với giá trị kích thước nền khác nhau cho mỗi hình ảnh:

4. Hình nền kích thước đầy đủ

html {  
  background: url(img\_man.jpg) no-repeat center fixed;  
  background-size: cover;  
}

**2.3.2.4 màu sắc**

1. Màu RGBA

Giá trị màu RGBA là phần mở rộng của giá trị màu RGB với kênh alpha - chỉ định độ mờ cho màu.

Giá trị màu RGBA được chỉ định bằng: rgba (đỏ, lục, lam, alpha). Tham số alpha là một số trong khoảng 0,0 (hoàn toàn trong suốt) và 1,0 (hoàn toàn mờ)

2. Màu sắc HSL

HSL là viết tắt của Hue, Saturation và Lightness.

Giá trị màu HSL được chỉ định bằng: hsl (màu sắc, độ bão hòa, độ sáng).

Hue là một độ trên bánh xe màu (từ 0 đến 360):

· 0 (hoặc 360) là màu đỏ

· 120 là màu xanh lá cây

· 240 là màu xanh

Độ bão hòa là một giá trị phần trăm: 100% là màu đầy đủ.

Độ nhẹ cũng là một tỷ lệ phần trăm; 0% là màu tối (màu đen) và 100% là màu trắng.

3. Màu sắc HSLA

Giá trị màu HSLA là phần mở rộng của giá trị màu HSL với kênh alpha - chỉ định độ mờ cho màu.

Giá trị màu HSLA được chỉ định bằng: hsla (màu sắc, độ bão hòa, độ sáng, alpha), trong đó tham số alpha xác định độ mờ. Tham số alpha là một số trong khoảng 0,0 (hoàn toàn trong suốt) và 1,0 (hoàn toàn mờ).

4. Độ mờ đục

opacityThuộc tính CSS đặt độ mờ cho toàn bộ thành phần (cả màu nền và văn bản sẽ mờ / trong suốt).

Các opacitygiá trị tài sản phải là một số giữa 0.0 (hoàn toàn trong suốt) và 1.0 (hoàn toàn mờ đục).

rgb (255, 0, 0); độ mờ: 0,2;

***2.3.2.4. CSS Gradients***

Độ dốc CSS cho phép bạn hiển thị các hiệu ứng chuyển tiếp mượt mà giữa hai hoặc nhiều màu được chỉ định.

CSS định nghĩa hai loại độ dốc:

Độ dốc tuyến tính (đi xuống / lên / trái / phải / chéo)

Radial Gradents (được xác định bởi trung tâm của họ)

1.CSS tuyến tính

Để tạo một gradient tuyến tính, bạn phải xác định ít nhất hai điểm dừng màu. Điểm dừng màu là màu bạn muốn hiển thị chuyển tiếp mượt mà giữa. Bạn cũng có thể đặt điểm bắt đầu và hướng (hoặc góc) cùng với hiệu ứng chuyển màu.

Cú pháp

background-image: linear-gradient(direction, color-stop1, color-stop2, ...);

Gradient tuyến tính - Từ trên xuống dưới (đây là mặc định)

Gradient tuyến tính - từ trái sang phải

Gradient tuyến tính - Đường chéo

2.Sử dụng góc

Nếu bạn muốn kiểm soát nhiều hơn về hướng của gradient, bạn có thể xác định một góc, thay vì các hướng được xác định trước (từ dưới lên, trên cùng, sang phải, sang trái, xuống dưới bên phải, v.v.).

Cú pháp

background-image: linear-gradient(angle, color-stop1, color-stop2);

Góc được chỉ định là một góc giữa một đường ngang và đường dốc.

3.Sử dụng nhiều màu dừng

4.Sử dụng tính minh bạch

Độ dốc CSS cũng hỗ trợ độ trong suốt, có thể được sử dụng để tạo hiệu ứng mờ dần.

Để thêm độ trong suốt, chúng tôi sử dụng hàm rgba () để xác định các điểm dừng màu. Tham số cuối cùng trong hàm rgba () có thể là giá trị từ 0 đến 1 và nó xác định độ trong suốt của màu: 0 biểu thị độ trong suốt đầy đủ, 1 biểu thị màu đầy đủ (không trong suốt).

5.Lặp lại một gradient tuyến tính

Hàm lặp lại tuyến tính-gradient () được sử dụng để lặp lại độ dốc tuyến tính

6.CSS Radial Gradents

Một gradient xuyên tâm được xác định bởi trung tâm của nó.

Để tạo một gradient xuyên tâm, bạn cũng phải xác định ít nhất hai điểm dừng màu.

Cú pháp

background-image: radial-gradient(shape size at position, start-color, ..., last-color);

Theo mặc định, hình dạng là hình elip, kích thước là góc xa nhất và vị trí là trung tâm.

Radial Gradient - Dừng màu cách đều nhau (đây là mặc định)

Radial Gradient - Dừng màu cách đều nhau

7.Đặt hình

Tham số hình dạng xác định hình dạng. Nó có thể lấy vòng tròn giá trị hoặc hình elip. Giá trị mặc định là hình elip.

8.Sử dụng các từ khóa kích thước khác nhau

Tham số kích thước xác định kích thước của gradient. Nó có thể mất bốn giá trị:

gần nhất

phía xa nhất

góc gần nhất

góc xa nhất

9.Lặp lại một gradient xuyên tâm

Hàm lặp lại-radial-gradient () được sử dụng để lặp lại độ dốc xuyên tâm:

***2.3.2.5 CSS Gradients***

Độ dốc CSS cho phép bạn hiển thị các hiệu ứng chuyển tiếp mượt mà giữa hai hoặc nhiều màu được chỉ định.

CSS định nghĩa hai loại độ dốc:

Độ dốc tuyến tính (đi xuống / lên / trái / phải / chéo)

Radial Gradents (được xác định bởi trung tâm của họ)

1.CSS tuyến tính

Để tạo một gradient tuyến tính, bạn phải xác định ít nhất hai điểm dừng màu. Điểm dừng màu là màu bạn muốn hiển thị chuyển tiếp mượt mà giữa. Bạn cũng có thể đặt điểm bắt đầu và hướng (hoặc góc) cùng với hiệu ứng chuyển màu.

Cú pháp

background-image: linear-gradient(direction, color-stop1, color-stop2, ...);

2.Sử dụng góc

Nếu bạn muốn kiểm soát nhiều hơn về hướng của gradient, bạn có thể xác định một góc, thay vì các hướng được xác định trước (từ dưới lên, trên cùng, sang phải, sang trái, xuống dưới bên phải, v.v.).

Cú pháp

background-image: linear-gradient(angle, color-stop1, color-stop2);

Góc được chỉ định là một góc giữa một đường ngang và đường dốc.

3.Sử dụng tính minh bạch

Độ dốc CSS cũng hỗ trợ độ trong suốt, có thể được sử dụng để tạo hiệu ứng mờ dần.

Để thêm độ trong suốt, chúng tôi sử dụng hàm rgba () để xác định các điểm dừng màu. Tham số cuối cùng trong hàm rgba () có thể là giá trị từ 0 đến 1 và nó xác định độ trong suốt của màu: 0 biểu thị độ trong suốt đầy đủ, 1 biểu thị màu đầy đủ (không trong suốt).

4.CSS Radial Gradents

Một gradient xuyên tâm được xác định bởi trung tâm của nó.

Để tạo một gradient xuyên tâm, bạn cũng phải xác định ít nhất hai điểm dừng màu.

Cú pháp

background-image: radial-gradient(shape size at position, start-color, ..., last-color);

Theo mặc định, hình dạng là hình elip, kích thước là góc xa nhất và vị trí là trung tâm.

5.Đặt hình

Tham số hình dạng xác định hình dạng. Nó có thể lấy vòng tròn giá trị hoặc hình elip. Giá trị mặc định là hình elip

6.Sử dụng các từ khóa kích thước khác nhau

Tham số kích thước xác định kích thước của gradient. Nó có thể mất bốn giá trị:

gần nhất

phía xa nhất

góc gần nhất

góc xa nhất

7. Lặp lại một gradient xuyên tâm

Hàm lặp lại-radial-gradient () được sử dụng để lặp lại độ dốc xuyên tâm

***2.3.2.6 CSS Shadows***

1.Hiệu ứng bóng CSS

Với CSS, bạn có thể thêm bóng cho văn bản và các phần tử.

Trong chương này, bạn sẽ tìm hiểu về các thuộc tính sau:

text-shadow

box-shadow

2.Bóng văn bản CSS

text-shadowThuộc tính CSS áp dụng bóng cho văn bản.

Trong cách sử dụng đơn giản nhất, bạn chỉ chỉ định bóng ngang (2px) và bóng dọc (2px)

3.Nhiều bóng

Để thêm nhiều bóng vào văn bản, bạn có thể thêm danh sách bóng được phân tách bằng dấu phẩy.

4.Thuộc tính hộp bóng CSS

CSS box-shadowtài sản áp dụng bóng tối để yếu tố này.

Trong cách sử dụng đơn giản nhất, bạn chỉ xác định bóng ngang và bóng dọc

5.Thuộc tính bóng CSS

Bảng sau liệt kê các thuộc tính bóng CSS

Property | Description

box-shadow | Adds one or more shadows to an element

text-shadow | Adds one or more shadows to a text

***2.3.2.7 CSS Text Effects***

1.Tràn văn bản CSS , Word Word, Quy tắc ngắt dòng và n sẽ tìChế độ viết

Trong chương này, bạm hiểu về các thuộc tính sau:

text-overflow

word-wrap

word-break

writing-mode

2.Tràn văn bản CSS

text-overflowThuộc tính CSS chỉ định cách thức nội dung tràn không được hiển thị sẽ được báo hiệu cho người dùng.

Nó có thể được cắt bớt

3.CSS Word Wrapping

word-wrapThuộc tính CSS cho phép các từ dài có thể bị phá vỡ và gói vào dòng tiếp theo.

Nếu một từ quá dài để vừa trong một khu vực, nó sẽ mở rộng ra bên ngoài

Thuộc tính bao bọc từ cho phép bạn buộc văn bản phải bọc - ngay cả khi nó có nghĩa là tách nó ở giữa một từ

4.CSS Word Breaking

word-breakThuộc tính CSS chỉ định quy tắc ngắt dòng

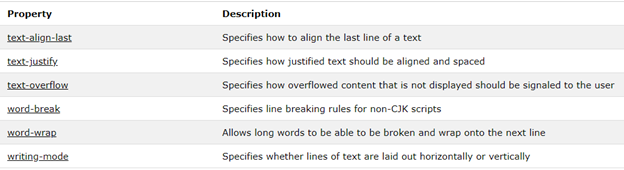
5.Chế độ viết CSS

writing-modeThuộc tính CSS chỉ định liệu các dòng văn bản được đặt theo chiều ngang hay chiều dọc.

Một số văn bản có thành phần span với chế độ viết

6.Thuộc tính hiệu ứng văn bản CSS

Bảng sau liệt kê các thuộc tính hiệu ứng văn bản CSS

**

***2.3.2.8 CSS Web Fonts***

1. Tràn văn bản CSS , Word Word, Quy tắc ngắt dòng và Chế độ viết

Trong chương này, bạn sẽ tìm hiểu về các thuộc tính sau:

text-overflow

word-wrap

word-break

writing-mode

2. Tràn văn bản CSS

text-overflowThuộc tính CSS chỉ định cách thức nội dung tràn không được hiển thị sẽ được báo hiệu cho người dùng.

Nó có thể được cắt bớt

3. CSS Word Wrapping

word-wrapThuộc tính CSS cho phép các từ dài có thể bị phá vỡ và gói vào dòng tiếp theo.

Nếu một từ quá dài để vừa trong một khu vực, nó sẽ mở rộng ra bên ngoài

Thuộc tính bao bọc từ cho phép bạn buộc văn bản phải bọc - ngay cả khi nó có nghĩa là tách nó ở giữa một từ

4. CSS Word Breaking

word-breakThuộc tính CSS chỉ định quy tắc ngắt dòng

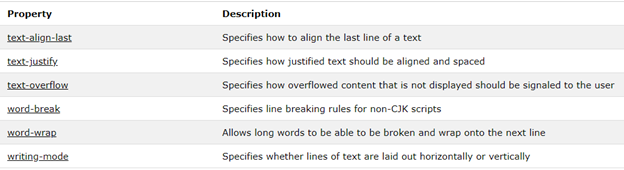
5. Chế độ viết CSS

writing-modeThuộc tính CSS chỉ định liệu các dòng văn bản được đặt theo chiều ngang hay chiều dọc.

Một số văn bản có thành phần span với chế độ viết

6. Thuộc tính hiệu ứng văn bản CSS

Bảng sau liệt kê các thuộc tính hiệu ứng văn bản CSS



1. Quy tắc khuôn mặt CSS @ phông chữ

Phông chữ web cho phép các nhà thiết kế web sử dụng các phông chữ không được cài đặt trên máy tính của người dùng.

Khi bạn đã tìm thấy / mua phông chữ bạn muốn sử dụng, chỉ cần bao gồm tệp phông chữ trên máy chủ web của bạn và nó sẽ được tự động tải xuống cho người dùng khi cần.

Phông chữ "của riêng bạn" được xác định trong @font-facequy tắc CSS .

2.Định dạng phông chữ khác nhau

Phông chữ TrueType (TTF)

TrueType là một tiêu chuẩn phông chữ được phát triển vào cuối những năm 1980, bởi Apple và Microsoft. TrueType là định dạng phông chữ phổ biến nhất cho cả hệ điều hành Mac OS và Microsoft Windows.

Phông chữ OpenType (OTF)

OpenType là một định dạng cho phông chữ máy tính có thể mở rộng. Nó được xây dựng trên TrueType và là nhãn hiệu đã đăng ký của Microsoft. Phông chữ OpenType được sử dụng phổ biến hiện nay trên các nền tảng máy tính chính.

Định dạng phông chữ mở trên web (WOFF)

WOFF là một định dạng phông chữ để sử dụng trong các trang web. Nó được phát triển vào năm 2009 và hiện là Khuyến nghị của W3C. WOFF về cơ bản là OpenType hoặc TrueType với nén và siêu dữ liệu bổ sung. Mục tiêu là hỗ trợ phân phối phông chữ từ máy chủ đến máy khách qua mạng với các ràng buộc về băng thông.

Định dạng phông chữ mở trên web (WOFF 2.0)

Phông chữ TrueType / OpenType cung cấp khả năng nén tốt hơn WOFF 1.0.

Phông chữ SVG / Hình dạng

Phông chữ SVG cho phép SVG được sử dụng làm glyphs khi hiển thị văn bản. Đặc tả SVG 1.1 xác định mô-đun phông chữ cho phép tạo phông chữ trong tài liệu SVG. Bạn cũng có thể áp dụng CSS cho các tài liệu SVG và quy tắc khuôn mặt @ phông chữ có thể được áp dụng cho văn bản trong tài liệu SVG.

Phông chữ OpenType nhúng (EOT)

Phông chữ EOT là một dạng phông chữ OpenType nhỏ gọn được Microsoft thiết kế để sử dụng làm phông chữ nhúng trên các trang web.

3.Hỗ trợ trình duyệt cho định dạng phông chữ

Các số trong bảng chỉ định phiên bản trình duyệt đầu tiên hỗ trợ đầy đủ định dạng phông chữ.

4.Sử dụng phông chữ bạn muốn

Trong @font-facequy tắc; đầu tiên xác định tên cho phông chữ (ví dụ myFirstFont) và sau đó trỏ đến tệp phông chữ.Để sử dụng phông chữ cho một phần tử HTML, hãy tham khảo tên của phông chữ (myFirstFont) thông qua thuộc font-familytính

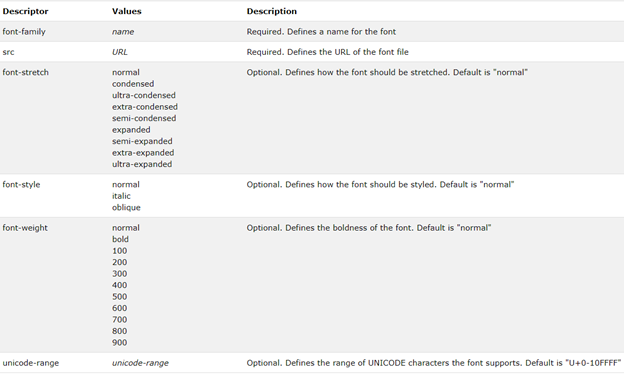
5. Sử dụng chữ đậm

Bạn phải thêm một @font-facequy tắc khác có chứa các mô tả cho văn bản in đậm

Tệp "sansation\_bold.woff" là một tệp phông chữ khác, chứa các ký tự in đậm cho phông chữ Sansation.

Các trình duyệt sẽ sử dụng điều này bất cứ khi nào một đoạn văn bản có họ phông chữ "myFirstFont" sẽ được in đậm.

Bằng cách này bạn có thể có nhiều @font-facequy tắc cho cùng một phông chữ

*. Mô tả phông chữ CSS*

**2.3.2.9 CSS 2D Transforms**

2D Transforms được sử dụng để tái thay đổi cấu trúc phần tử, ví dụ như translate, rotate, scale, và skew

Bảng dưới liệt kê một số giá trị thường được sử dụng trong 2D Transforms:

|  |  |
| --- | --- |
| ***Giá trị*** | ***Miêu tả*** |
| *matrix(n,n,n,n,n,n)* | *Được sử dụng để định nghĩa Matrix Transforms (dạng tịnh tiến theo ma trận) với 6 giá trị* |
| *translate(x,y)* | *Được sử dụng để tịnh tiến phần tử theo trục x và trục y* |
| *translateX(n)* | *Được sử dụng để tịnh tiến phần tử theo trục x* |
| *translateY(n)* | *Được sử dụng để tịnh tiến phần tử theo trục y* |
| *scale(x,y)* | *Được sử dụng để thay đổi độ rộng và chiều cao của phần tử* |
| *scaleX(n)* | *Được sử dụng để thay đổi độ rộng của phần tử* |
| *scaleY(n)* | *Được sử dụng để thay đổi chiều cao của phần tử* |
| *rotate(angle)* | *Được sử dụng để quay phần tử dựa trên một góc (angle)* |
| *skewX(angle)* | *Được sử dụng để định nghĩa Skew Transforms (dạng tịnh tiến đối xứng lệch) cùng với trục x* |
| *skewY(angle)* | *Được sử dụng để định nghĩa Skew Transforms cùng với trục y* |

**2.3.2.10 CSS 3D Transforms**

**Tiền tố cụ thể của trình duyệt**

*Một số trình duyệt cũ hơn cần tiền tố cụ thể (-webkit-) để hiểu các thuộc tính chuyển đổi 3D:*

*Thí dụ*

*#myDiv {*

*-webkit-transform: rotateY(130deg);/\* Safari prior 9.0 \*/*

*transform: rotateY(130deg);/\* Standard syntax \*/*

*}*

**Phương thức rotationY ()**

*Các rotateY()phương pháp quay một yếu tố xung quanh trục Y của nó ở một mức độ nhất định:*

*vd*

*#myDiv {*

*transform: rotateY(130deg);*

*}*

**Phương thức rotationZ ()**

*Các rotateZ()phương pháp quay một yếu tố xung quanh trục Z của nó ở một mức độ*

*#myDiv {*

*transform: rotateZ(90deg);*

*}*

**Thuộc tính chuyển đổi CSS**

Bảng sau liệt kê tất cả các thuộc tính biến đổi 3D:

|  |  |
| --- | --- |
| **Property** | **Description** |
| [transform](https://www.w3schools.com/cssref/css3_pr_transform.asp) | Áp dụng chuyển đổi 2D hoặc 3D cho một phần tử |
| [transform-origin](https://www.w3schools.com/cssref/css3_pr_transform-origin.asp) | Allows you to change the position on transformed elements |
| [transform-style](https://www.w3schools.com/cssref/css3_pr_transform-style.asp) | Specifies how nested elements are rendered in 3D space |
| [perspective](https://www.w3schools.com/cssref/css3_pr_perspective.asp) | Specifies the perspective on how 3D elements are viewed |
| [perspective-origin](https://www.w3schools.com/cssref/css3_pr_perspective-origin.asp) | Specifies the bottom position of 3D elements |
| [backface-visibility](https://www.w3schools.com/cssref/css3_pr_backface-visibility.asp) | Defines whether or not an element should be visible when not facing the screen |

**2.3.2.11 CSS Transitions**

Thuộc tính Transition trong CSS được sử dụng khá rộng rãi trong thiết kế web để tạo ra các hiệu ứng chuyển đổi đẹp mắt trên website một cách dễ dàng.

Transition hoạt động bằng cách thay đổi giá trị thuộc tính một cách trơn tru từ giá trị này sang giá trị khác trong khoảng thời gian nhất định. Các tham số thường được sử dụng:

• transition-delay: khoảng thời gian dừng cho mỗi hiệu ứng chuyển đổi.

• transition-duration: khoảng thời gian chuyển đổi diễn ra.

• transition-property: thuộc tính cần chuyển đổi.

• transition-timing-function: tốc độ chuyển đổi diễn ra.

Cách sử dụng Transition trong CSS

Để tạo ra hiệu ứng chuyển đổi Transition, bạn phải xác định ít nhất hai điều:

• Thuộc tính CSS muốn thêm hiệu ứng vào

• Thời gian chuyển đổi diễn ra (duration).

Lưu ý: Nếu phần duration không được chỉ định, quá trình chuyển đổi sẽ không diễn ra mượt mà, trơn tru vì giá trị mặc định bằng 0.

**Thay đổi giá trị một thuộc tính**

Ví dụ: Phần tử <div> màu tím 100px \* 100px, ta chỉ định hiệu ứng Transition cho thuộc tính width, với thời lượng 2 giây:

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<style>**
5. div {
6. width: 100px;
7. height: 100px;
8. background: purple;
9. -webkit-transition: width 2s; /\* For Safari 3.1 to 6.0 \*/
10. transition: width 2s;
11. }
13. div:hover {
14. width: 300px;
15. }
16. **</style>**
17. **</head>**
18. **<body>**
20. **<h1>**Thuộc tính transition**</h1>**
22. **<p>**Di chuột vào phần tử để thấy hiệu ứng transition:**</p>**
23. **<div></div>**
25. **</body>**
26. **</html>**

Lưu ý: Khi con trỏ di chuyển ra khỏi phần tử, nó sẽ dần thay đổi trở lại kiểu ban đầu.

Thay đổi giá trị nhiều thuộc tính

Ví dụ: Thêm hiệu ứng chuyển tiếp cho cả thuộc tính width và height, với thời lượng 2 giây cho chiều rộng và 4 giây cho chiều cao.

div {

-webkit-transition: width 2s, height 4s; /\* Safari \*/

transition: width 2s, height 4s;

}

**Tốc độ chuyển đổi**

Thuộc tính transition-timing-function dùng để xác định tốc độ thay đổi khi chuyển đổi.

Các giá trị có sẵn như sau:

• ease: tạo hiệu ứng chuyển đổi khi bắt đầu thì chậm sau đó nhanh dần và gần kết thúc lại chậm từ từ (giá trị mặc định).

• linear: tạo hiệu ứng chuyển đổi từ lúc bắt đầu với lúc kết thúc tốc độ là như nhau.

• ease-in: tạo hiệu ứng chuyển đổi chậm ở lúc bắt đầu.

• ease-out: tạo hiệu ứng chuyển đổi chậm ở lúc kết thúc.

• ease-in-out: tạo hiệu ứng chuyển đổi chậm ở lúc bắt đầu và lúc kết thúc.

#div1 {transition-timing-function: linear;}

#div2 {transition-timing-function: ease;}

#div3 {transition-timing-function: ease-in;}

#div4 {transition-timing-function: ease-out;}

#div5 {transition-timing-function: ease-in-out;}

**Độ trễ của hiệu ứng Transition**

Thuộc tính transition-delay sử dụng để xác định khoảng thời gian trì hoãn giữa thời gian một thuộc tính thay đổi và lúc chuyển tiếp thực sự bắt đầu.

Ví dụ: Độ trễ 1 giây trước khi bắt đầu chuyển đổi.

div {

-webkit-transition-delay: 1s; /\* Safari \*/

transition-delay: 1s;

}

hi rê chuột vào thì hiệu ứng không diễn ra ngay mà sẽ bị delay 1s là thời gian mà chúng ta đặt cho nó.

**Kết hợp Transition với Transform**

Transform Quantrimang đã tìm hiểu ở bài học trước, các bạn vào xem lại 2D Transform ở đây và 3D Transform ở đây nhé.

Kết hợp Transition với Transform với nhau sẽ tạo nên hiệu ứng chuyển tiếp cực kỳ đẹp mắt.

div {

-webkit-transition: width 2s, height 2s, -webkit-transform 2s; /\* Safari \*/

transition: width 2s, height 2s, transform 2s

**2.3.2.12 CSS Animations**

Animation được hiểu là hiệu ứng chuyển động, sử dụng để tạo hiệu ứng di chuyển cho các phần tử và được ứng dụng khá nhiều trong các website hiện nay.

Để tạo một chuyển động Animation, bạn cần phải có các keyframe. Mỗi keyframe sẽ được chạy ở một thời điểm xác định và trong keyframe đó nó quy định việc phần tử sẽ di chuyển ra sao.

Ngoài ra, Animation còn gồm một số thuộc tính quy định các chi tiết khá quan trọng của hiệu ứng thường đi kèm như:

• Thuộc tính animation-name

• Thuộc tính animation-duration

• Thuộc tính animation-timing-function

• Thuộc tính animation-delay

• Thuộc tính animation-iteration-count

• Thuộc tính aniamtion-direction

• Thuộc tính animation-fill-mode

Cùng Quantrimang.com tìm hiểu kỹ hơn về keyframe và các thuộc tính cần thiết trong nội dung tiếp theo.

**Quy tắc Keyframe**

Bên trong quy tắc này, bạn xác định các keyframe để quy định việc phần tử sẽ chuyển động ra sao tại mỗi thời điểm nhất định.

Cú pháp của keyframe:

@keyframes Name {

/\*code\*/

}

• Name: tên của animation bạn muốn tạo.

• code: Các đoạn code quy định tiến trình chuyển động. Có 2 dạng:

• Sử dụng phần trăm từ 0% đến 100%.

• from...to: thiết lập giá trị từ khởi đầu (from - tương đương với 0%) đến kết thúc (to - tương đương với 100%).

Để chuyển động xảy ra cần phải kết nối @keyframes với phần tử.

Lưu ý: Để tạo ra hiệu ứng Animation, bạn phải xác định ít nhất hai điều:

• Thuộc tính animation-duration là khoảng thời gian diễn ra hiệu ứng. Nếu phần duration không được chỉ định sẽ không xảy ra hiệu ứng vì giá trị mặc định bằng 0.

• Thuộc tính animation-name xác định phần tử sẽ thực thi animation nào.

**Thuộc tính animation-delay**

Thuộc tính animation-delay sử dụng để xác định khoảng thời gian trì hoãn giữa thời gian một thuộc tính thay đổi và lúc hiệu ứng animation thực sự bắt đầu.

**Thuộc tính animation-iteration-count**

Thuộc tính animation-iteration-count sử dụng để thiết lập số lần thực hiện một animation. Giá trị thường là:

• Một số lần nhất định

• infinite: animation lặp lại liên tục và vô hạn

**Thuộc tính animation-direction**

Thuộc tính animation-direction sử dụng để xác định chiều chạy của animation. Các giá trị mà animation-direction có thể nhận là:

• normal: animation di chuyển bình thường tiến về phía trước (mặc định)

• reverse: animation di chuyển theo hướng ngược lại, lui về sau.

• alternate: animation di chuyển tiến về trước, sau đó lui theo hướng ngược lại

• alternate-reverse: animation di chuyển ngược lại trước, rồi đổi chiều tiến về trước.

**Thuộc tính animation-timing-function**

Thuộc tính animation-timing-function dùng để xác định tốc độ thay đổi khi hiệu ứng di chuyển.

Các giá trị có sẵn như sau:

• ease: tạo hiệu ứng chuyển đổi khi bắt đầu thì chậm sau đó nhanh dần và gần kết thúc lại chậm dần (giá trị mặc định).

• linear: tạo hiệu ứng chuyển đổi từ lúc bắt đầu với lúc kết thúc tốc độ là như nhau.

• ease-in: tạo hiệu ứng chuyển đổi chậm lúc bắt đầu.

• ease-out: tạo hiệu ứng chuyển đổi chậm lúc kết thúc.

• ease-in-out: tạo hiệu ứng chuyển đổi chậm cả lúc bắt đầu và kết thúc.

• cubic-bezier(n,n,n,n): cho phép bạn xác định một giá trị của riêng mình theo hàm bezier

**Thuộc tính animation-fill-mode**

Animation CSS không gây ảnh hưởng đến phần tử trước khi chạy keyframe đầu tiên và sau khi keyframe cuối cùng kết thúc. Và thuộc tính animation-fill-mode sử dụng để thay đổi trạng thái của phần tử trước khi bắt đầu sau khi kết thúc Animation.

Các giá trị có sẵn như sau:

• none: khi animation không hoạt động thì nó sẽ giữ nguyên trạng thái bất động của phần tử, không thêm một style nào vào thành phần (mặc định).

• forwards: khi animation không hoạt động sau khi kết thúc animation, giá trị này sẽ apply các thuộc tính của lần cuối cùng xuất hiện trong keyframe vào trạng thái của phần tử (phụ thuộc vào animation-direction và animation-iteration-count).

• backwards: khi animation không hoạt động trước khi bắt đầu animation (đang trong thời gian delay), giá trị này sẽ apply các thuộc tính của lần xuất hiện đầu tiên trong keyfame vào trạng thái của phần tử (phụ thuộc vào thuộc tính anmation-direction).

• both: kết hợp cả forwards và backwards cho trạng thái phần tử.

• Gộp chung các thuộc tính

• Ta có một animation khai báo đầy đủ 6 thuộc tính như sau:

• div {

animation-name: example;

animation-duration: 5s;

animation-timing-function: linear;

animation-delay: 2s;

animation-iteration-count: infinite;

animation-direction: alternate;

}

• Tuy nhiên trong một vài trường hợp, việc khai đầy đủ như trên là không cần thiết và dài dòng. Vì vậy CSS hỗ trợ chúng ta một thuộc tính có thể khai báo toàn bộ giá trị của các thuộc tính trên, đó là thuộc tính animation.

• Cú pháp sử dụng như sau (chú ý thứ tự khai báo):

animation: name | duration | timing-function | delay | iteration-count | direction | fill-mode

• Vậy ví dụ trên có thể được khai báo ngắn gọn trong 1 dòng như sau:

div {

animation: example 5s linear 2s infinite alternate;

**2.3.2.13 CSS tooltips**

**Tooltip** thường được sử dụng để làm xuất hiện một hộp nhỏ khi di chuột qua văn bản hoặc hình ảnh nào đó với thông tin liên quan đến thành phần được chuột di qua. Xét về mặt trải nghiệm người dùng, các tooltip cung cấp cho người dùng nguồn thông tin một cách nhanh chóng và dễ dàng nhất mà không phải nhấp vào bất kỳ thứ gì.

ví dụ:

1. <!DOCTYPE html>
2. <html>
3. <style>
4. .tooltip {
5. **position**: relative;
6. **display**: inline-block;
7. **border-bottom**: 1px dotted black;}
8. .tooltip .tooltiptext {
9. **visibility**: hidden;
10. **width**: 120px;
11. **background-color**: black;
12. **color**: #fff;
13. **text-align**: center;
14. border-radius: 6px;
15. **padding**: 5px 0;
16. /\* Position the tooltip \*/
17. **position**: absolute;
18. **z-index**: 1;}
20. .tooltip:hover .tooltiptext {
21. **visibility**: visible;}
22. </style>
23. <body style="text-align:center;">
24. <p>Di chuột qua văn bản dưới đây:</p>
25. <div class="tooltip">Hãy di chuột qua tôi nào!
26. <span class="tooltiptext">Tooltip text</span>
27. </div></body>
28. </html>

**Vị trí xuất hiện của Tooltip**

*Tooltip nằm ở bên cạnh (Trái hoặc phải)*

.tooltip .tooltiptext {

top: -5px;

left: 105%;

}

Trong ví dụ này, *Tooltip được đặt ở bên phải* (left: 105%) của đoạn text "Hãy di chuột qua tôi nào!" (<div>).

Lưu ý rằng top: -5px được sử dụng để đặt Tooltip **ở giữa phần tử chứa nó** , vì tooltip có phần padding top và padding bottom là 5px. Nếu bạn tăng giá trị này thì cũng nên tăng giá trị top để luôn đảm bảo rằng tooltip nằm ở giữa (cho cân đối, đẹp mắt). Điều tương tự cũng áp dụng nếu bạn muốn tooltip đặt ở bên trái.

Tương tự với *Tooltip đặt ở bên trái.*

.tooltip .tooltiptext {

top: -5px;

right: 105%;}

*Tooltip nằm ở phía dưới*

1. tooltip .tooltiptext {
2. **width**: 120px;
3. top: 100%;
4. left: 50%;
5. **margin-left**: -60px; /\* Sử dụng một nửa chiều rộng (120/2 = 60), để căn giữa tooltip \*/
6. }

Một mũi tên, móc nhọn gắn với Tooltip làm chúng trở nên sinh động hơn, giống một trích dẫn lời nói chân thực

**Lưu ý:** Thuộc tính border-width chỉ định kích thước của mũi tên, nếu bạn thay giá trị này thì cũng thay đổi giá trị margin-left thành giống như thế để mũi tên luôn nằm ở giữa.

Tương tự với *mũi tên của Tooltip nằm bên dưới:*

1. .tooltip .tooltiptext::after {
2. **content**: " ";
3. **position**: absolute;
4. bottom: 100%; /\* mũi tên ở phía trên của tooltip \*/
5. left: 50%;
6. **margin-left**: -5px;
7. **border-width**: 5px;
8. **border-style**: solid
9. **border-color**: transparent transparent black transparent;

**2.3.2.14 CSS Style Images**

**Hình ảnh tròn**

Sử dụng thuộc border-radiustính để tạo hình ảnh tròn:

Thí dụ

Hình ảnh tròn:

img {

border-radius: 8px;

}

**Hình ảnh thu nhỏ**

Sử dụng bordertài sản để tạo hình ảnh thu nhỏ.

Hình ảnh thu nhỏ:

Thí dụ

1. img {
2. **border**: 1px solid #ddd;
3. border-radius: 4px;
4. **padding**: 5px;
5. **width**: 150px;
6. }
8. <img src="paris.jpg" alt="Paris">

**Hình ảnh phản hồi**

Hình ảnh phản hồi sẽ tự động điều chỉnh để phù hợp với kích thước của màn hình.

Thay đổi kích thước cửa sổ trình duyệt để xem hiệu ứng:

Nếu bạn muốn một hình ảnh thu nhỏ lại nếu nó phải, nhưng không bao giờ tăng tỷ lệ lớn hơn kích thước ban đầu của nó, hãy thêm vào như sau:

Thí dụ

img {

max-width: 100%;

height: auto;

}

**Căn giữa một hình ảnh**

Để căn giữa một hình ảnh, đặt lề trái và phải thành autovà biến nó thành một thành blockphần:

Thí dụ

1. img {
2. **display**: block;
3. **margin-left**: auto;
4. **margin-right**: auto;
5. **width**: 50%;
6. }

**Hình ảnh / Thẻ Polaroid**

Thí dụ

1. div.polaroid {
2. **width**: 80%;
3. **background-color**: white;
4. box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);
5. }
7. img {**width**: 100%}
9. div.container {
10. **text-align**: center;
11. **padding**: 10px 20px;
12. }

**Hình ảnh trong suốt**

Tài opacitysản có thể lấy giá trị từ 0,0 - 1,0.

Lưu ý: IE8 và sử dụng trước đó filter:alpha(opacity=x). X có thể lấy giá trị từ 0 - 100. Giá trị thấp hơn làm cho phần tử trong suốt hơn.

Thí dụ

img {

opacity: 0.5;

filter: alpha(opacity=50); /\* For IE8 and earlier \*/

}

**Văn bản hình ảnh**

Cách định vị văn bản trong ảnh:

Vd :

1. <HTML>
2. **<head>**
3. **<style>**
4. .container {
5. position: relative;
6. }
8. .topleft {
9. position: absolute;
10. top: 8px;
11. left: 16px;
12. font-size: 18px;
13. }
15. img {
16. width: 100%;
17. height: auto;
18. opacity: 0.3;
19. }
20. **</style>**
21. **</head>**
22. **<body>**
24. **<h2>**Image Text**</h2>**
25. **<p>**Add some text to an image in the top left corner:**</p>**
27. **<div** class="container"**>**
28. **<img** src="img\_5terre\_wide.jpg" alt="Cinque Terre" width="1000" height="300"**>**
29. **<div** class="topleft"**>**Top Left**</div>**
30. **</div>**
32. **</body>**
33. **</html>**

**Bộ lọc hình ảnh**

filterThuộc tính CSS thêm hiệu ứng hình ảnh (như mờ và bão hòa) cho một yếu tố.

Lưu ý: Thuộc tính bộ lọc không được hỗ trợ trong Internet Explorer, Edge 12 hoặc Safari 5.1 trở về trước.

Thí dụ

Thay đổi màu của tất cả các hình ảnh thành đen trắng (100% màu xám):

img {

-webkit-filter: grayscale(100%); /\* Safari 6.0 - 9.0 \*/

filter: grayscale(100%);

}

**Thư viện ảnh phản hồi**

CSS có thể được sử dụng để tạo các phòng trưng bày hình ảnh. Ví dụ này sử dụng truy vấn phương tiện để sắp xếp lại hình ảnh trên các kích thước màn hình khác nhau. Thay đổi kích thước cửa sổ trình duyệt để xem hiệu ứng:

Thí dụ

1. responsive {
2. **padding**: 0 6px;
3. **float**: left;
4. **width**: 24.99999%;
5. }
7. @media only screen and (**max-width**: 700px){
8. .responsive {
9. **width**: 49.99999%;
10. **margin**: 6px 0;
11. }
12. }
14. @media only screen and (**max-width**: 500px){
15. .responsive {
16. **width**: 100%;
17. }
18. }

***2.3.2.15 CSS Object-fit***

**object-fit** Thuộc tính CSS được sử dụng để chỉ định cách thay đổi kích thước <img> hoặc <video> để phù hợp với vùng chứa của nó.

Thuộc tính này cho biết nội dung để điền vào container theo nhiều cách khác nhau; chẳng hạn như "giữ tỷ lệ khung hình đó" hoặc "kéo dài và chiếm càng nhiều không gian càng tốt".

object-fit có thể có các giá trị sau:

• fill- Đây là mặc định. Nội dung thay thế có kích thước để điền vào hộp nội dung của phần tử. Nếu cần thiết, đối tượng sẽ được kéo dài hoặc cắt cho phù hợp

• contain - Nội dung được thay thế được thu nhỏ để duy trì tỷ lệ khung hình của nó trong khi vừa với hộp nội dung của phần tử

• cover- Nội dung được thay thế có kích thước để duy trì tỷ lệ khung hình của nó trong khi điền vào toàn bộ hộp nội dung của phần tử. Đối tượng sẽ được cắt bớt để phù hợp

• none - Nội dung thay thế không được thay đổi kích thước

• scale-down - Nội dung có kích thước như thể không có hoặc chứa được chỉ định (sẽ dẫn đến kích thước đối tượng cụ thể nhỏ hơn)

***2.3.2.16 CSS Buttons***

Tìm hiểu cách tạo kiểu cho các nút bằng CSS.

**Kiểu dáng nút cơ bản**

Thí dụ

1. .button {
2. **background-color**: #4CAF50; /\* Green \*/
3. **border**: none;
4. **color**: white;
5. **padding**: 15px 32px;
6. **text-align**: center;
7. **text-decoration**: none;
8. **display**: inline-block;
9. **font-size**: 16px;
10. }

**Màu nút**

Sử dụng thuộc background-color tính để thay đổi màu nền của nút:

Thí dụ

1. . button1 {**background-color**: #4CAF50;} /\* Green \*/
2. .button2 {**background-color**: #008CBA;} /\* Blue \*/
3. .button3 {**background-color**: #f44336;} /\* Red \*/
4. .button4 {**background-color**: #e7e7e7; **color**: black;} /\* Gray \*/
5. .button5 {**background-color**: #555555;} /\* Black \*/

**Kích thước nút**

Sử dụng thuộc font-size tính để thay đổi kích thước phông chữ của một nút:

Thí dụ

1. .button1 {**font-size**: 10px;}
2. .button2 {**font-size**: 12px;}
3. .button3 {**font-size**: 16px;}
4. .button4 {**font-size**: 20px;}
5. .button5 {**font-size**: 24px;}

**Sử dụng thuộc padding tính để thay đổi phần đệm của nút:**

Thí dụ

1. .button1 {**padding**: 10px 24px;}
2. .button2 {**padding**: 12px 28px;}
3. .button3 {**padding**: 14px 40px;}
4. .button4 {**padding**: 32px 16px;}
5. .button5 {**padding**: 16px;}

**Nút tròn**

Sử dụng thuộc border-radiustính để thêm các góc tròn vào một nút:

Thí dụ

.button1 {border-radius: 2px;}

.button2 {border-radius: 4px;}

.button3 {border-radius: 8px;}

.button4 {border-radius: 12px;}

.button5 {border-radius: 50%;}

**Sử dụng thuộc border tính để thêm viền màu vào nút:**

Thí dụ

1. .button1 {
2. **background-color**: white;
3. **color**: black;
4. **border**: 2px solid #4CAF50; /\* Green \*/
5. }

**Sử dụng :hover bộ chọn để thay đổi kiểu nút khi bạn di chuyển chuột qua nút đó.**

Mẹo: Sử dụng thuộc transition-durationtính để xác định tốc độ của hiệu ứng "di chuột":

Thí dụ

1. .button {
2. -webkit-transition-duration: 0.4s; /\* Safari \*/
3. transition-duration: 0.4s;
4. }
6. .button:hover {
7. **background-color**: #4CAF50; /\* Green \*/
8. **color**: white;
9. }
10. ...

**Sử dụng thuộc box-shadow tính để thêm bóng vào nút:**

Thí dụ

1. . button1 {
2. Box-shadow: 0 8px 16px 0 rgba(0,0,0,0.2), 0 6px 20px 0 rgba(0,0,0,0.19);
3. }
5. .button2:hover {
6. box-shadow: 0 12px 16px 0 rgba(0,0,0,0.24), 0 17px 50px 0 rgba(0,0,0,0.19);
7. }

**Sử dụng thuộctính opacityđể thêm độ trong suốt cho một nút (tạo giao diện "bị vô hiệu hóa").**

Mẹo: Bạn cũng có thể thêm thuộc cursortính có giá trị "không được phép", sẽ hiển thị "không có dấu hiệu đỗ xe" khi bạn di chuột qua nút:

Thí dụ

.disabled {

opacity: 0.6;

cursor: not-allowed;

Theo mặc định, kích thước của nút được xác định bởi nội dung văn bản của nó (rộng bằng nội dung của nó). Sử dụng width để thay đổi độ rộng của nút:

Thí dụ

.button1 {width: 250px;}

.button2 {width: 50%;}

.button3 {width: 100%;}

*Xóa lề và thêm float:left vào từng nút để tạo nhóm nút:*

Thí dụ

.button {

float: left;

}

**Sử dụng display:block thay vì float:left nhóm các nút bên dưới nhau, thay vì cạnh nhau:**

Thí dụ

.button {

display: block;

}

***2.3.2.17 CSS Pagination***

* + Simple Pagination

Nếu bạn có một trang web với rất nhiều trang, bạn có thể thêm một số loại phân trang cho mỗi trang:

.pagination {  
  display: inline-block;  
}  
  
.pagination a {  
  color: black;  
  float: left;  
  padding: 8px 16px;  
  text-decoration: none;  
}

* + Active and Hoverable Pagination

Đánh dấu trang hiện tại với một class .active, và sử dụng selector :hover để thay đổi màu sắc của mỗi liên kết trang khi di chuột qua chúng:

.pagination a.active {  
  background-color: #4CAF50;  
  color: white;  
}  
  
.pagination a:hover:not(.active) {background-color: #ddd;}

* + Rounded Active and Hoverable Buttons

Thêm thuộc tính border-radius nếu bạn muốn nút “active” và “hover” có hình tròn:

.pagination a.active {  
  background-color: #4CAF50;  
  color: white;  
}  
  
.pagination a:hover:not(.active) {background-color: #ddd;}

* + Hoverable Transition Effect

Thêm thuộc tính transition vào liên kết trang để tạo ra một hiệu ứng di chuột.

.pagination a {  
  transition: background-color .3s;  
}

* + Bordered Pagination

Sử dụng thuộc tính border để thêm khung viền cho phân trang:

.pagination a {  
  border: 1px solid #ddd; /\* Gray \*/  
}

* + Rounded Borders

Thêm viền tròn cho liên kết đầu tiên và cuối cùng của bạn trong phân trang:

.pagination a:first-child {  
  border-top-left-radius: 5px;  
  border-bottom-left-radius: 5px;  
}  
  
.pagination a:last-child {  
  border-top-right-radius: 5px;  
  border-bottom-right-radius: 5px;  
}

* + Space Betwween Links

Thêm thuộc tính margin nếu bạn không muốn nhóm các liên kết trang:

.pagination a {  
  margin: 0 4px; /\* 0 is for top and bottom. Feel free to change it \*/

[Try it Yourself »](https://www.w3schools.com/css/tryit.asp?filename=trycss_ex_pagination_margin)

* + Pagination Size

Thay đổi kích thước của phân trang với thuộc tính font-size :

.pagination a {  
  font-size: 22px;  
}

* + Centered Pagination

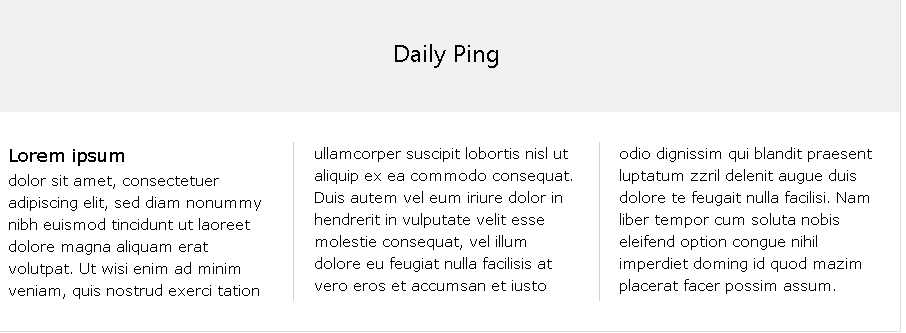
Để căn giữa phân trang, cuộn một phần tử container ( như <div>) xung quanh nó với text-align:center

.center {  
  text-align: center;  
}

***2.3.2.18 CSS Multiple Columns***

* + CSS Multi-column layout

Cách bố trí CSS nhiều cột cho phép dễ dàng định nghĩa nhiều cột của văn bản giống như trong tờ báo :



* + CSS Multi-column Properties

Các thuộc tính đa cột:

* Column-count
* Column-gap
* Column-rule-style
* Column-rule-width
* Column-rule-color
* Column-rule
* Column-span
* Column-width
* Các trình duyệt hỗ trợ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Property | Chrome | Internet Explorer | Firefox | Safari | Opera |
| column-count | 50.0 | 10.0 | 52.0 | 9.0 | 37.0 |
| column-gap | 50.0 | 10.0 | 52.0 | 9.0 | 37.0 |
| column-rule | 50.0 | 10.0 | 52.0 | 9.0 | 37.0 |
| column-rule-color | 50.0 | 10.0 | 52.0 | 9.0 | 37.0 |
| column-rule-style | 50.0 | 10.0 | 52.0 | 9.0 | 37.0 |
| column-rule-width | 50.0 | 10.0 | 52.0 | 9.0 | 37.0 |
| column-span | 50.0 | 10.0 | Not supported | 9.0 | 37.0 |
| column-width | 50.0 | 10.0 | 52.0 | 9.0 | 37.0 |

* + Tạo nhiều cột

Thuộc tính column-count xác định số cột của một phần tử nên được chia ra.

Ví dụ sau đây sẽ chia văn bản trong phần tử <div> thành 3 cột:

div {  
  column-count: 3;  
}

* + Xác định khoảng cách giữa các cột

Thuộc tính column-gap quy định khoảng cách giữa các cột.

div {  
  column-gap: 40px;  
}

* + Quy tắc cột trong CSS

Thuộc tính column-rule-style xác định kiểu của quy tắc giữa các cột :

div {  
  column-rule-style: solid;  
}

Thuộc tính column-rule-width quy định độ rộng giữa các cột.

div {  
  column-rule-width: 1px;  
}

Thuộc tính column-rule-color quy định màu sắc giữa các cột:

div {  
  column-rule-color: lightblue;  
}

Thuộc tính column-rule là một thuộc tính viết tắt để thiết lập tất cả các thuộc tính column-rule-\* bên trên.

Ví dụ sau thiết lập chiều rộng, kiểu và màu sắc của quy tắc giữa các cột:

div {  
  column-rule: 1px solid lightblue;  
}

* + Xác định bao nhiêu cột một phần tử nên kéo dài

Thuộc tính column-span xác định có bao nhiêu cột một phần tử nên kéo dài qua.

Ví dụ sau đây xác định rằng phần tử <h2> nên kéo dài qua tất cả các cột:

h2 {  
  column-span: all;  
}

* + Xác định chiều rộng cột

Thuộc tính column-width quy định cụ thể một gợi ý, chiều rộng tối ưu cho các cột.

Ví dụ dưới đây xác định các gợi ý, chiều rộng tối ưu cho các cột nên là 100px:

div {  
  column-width: 100px;  
}

***2.3.2.19 CSS User Interface***

* + CSS user interface

Các thuộc tính interface trong CSS:

* Resize
* Outline-offset
  + Các trình duyệt hỗ trợ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Property | Chrome | Internet Explorer | Firefox | Safari | Opera |
| resize | 4.0 | 15.0 | 5.0 | 4.0 | 15.0 |
| outline-offset | 4.0 | 15.0 | 5.0 | 4.0 | 9.5 |

* + CSS Resizing

Thuộc tính resize quy định nếu một phần tử nên thay đổi kích thước bởi người dùng.

div {  
  resize: horizontal;  
  overflow: auto;  
}

* + CSS outline offset

Thuộc tính outline-offset cho biết thêm không gian giữa một outline và cạnh hoắc đường viền của một phần tử.

Chú ý : outline khác với border! Không giống như border, outline được rút ra border của một phần tử, và có thể chồng chéo nội dung khác. Ngoài ra, outline không phải là một phần kích thước của phần tử; tổng chiều rộng và chiều cao của phần tử không ảnh hưởng đến độ rộng của outline.

Ví dụ sau đây sử dụng thuộc tính outline-offset để thêm khoảng trống giữa border và outline:

div.ex1 {  
  margin: 20px;  
  border: 1px solid black;  
  outline: 4px solid red;  
  outline-offset: 15px;  
}  
  
div.ex2 {  
  margin: 10px;  
  border: 1px solid black;  
  outline: 5px dashed blue;  
  outline-offset: 5px;  
}

* + CSS User Interface Properties

|  |  |
| --- | --- |
| **Thuộc tính** | **Mô tả** |
| [outline-offset](https://www.w3schools.com/cssref/css3_pr_outline-offset.asp) | Thêm khoảng trống giữa phần tử outline và edge hoặc border |
| [resize](https://www.w3schools.com/cssref/css3_pr_resize.asp) | Chỉ định xem người dùng có thể thay đổi kích thước phần tử hay không |

***2.3.2.20 CSS Variables***

* + CSS Custom Properties (Variables)

Hàm var() có thể sử dụng để chèn giá trị của thuộc tính tùy chỉnh.

* + Các trình duyệt hỗ trợ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function | Chrome | Internet Explorer | Firefox | Safari | Opera |
| var() | 49.0 | 15.0 | 31.0 | 9.1 | 36.0 |

* + Hàm var()

Các biến trong CSS phải được khai báo trong bộ chọn CSS nhằm xác định phạm vi của nó. Đối với phạm vi toàn thể, bạn có thể sử dụng :root hoặc body selector.

Tên của biến phải bắt đầu với hai dấu gạch ngang (--).

Cú pháp của hàm var() :

var(custom-name, value)

|  |  |
| --- | --- |
| **Giá trị** | **Mô tả** |
| *custom-name* | Bắt buộc . Tên thuộc tính tùy chỉnh (phải bắt đầu bằng hai dấu gạch ngang) |
| *value* | Tùy chọn. Giá trị dự phòng (được sử dụng khi thuộc tính tùy chỉnh không hợp lệ) |

Ví dụ sau đầu tiên định nghĩa một thuộc tính tùy chỉnh toàn cục có tên là “—main-bg-color”, sau đó sử dụng hàm var() để chèn giá trị thuộc tính riêng sau đó trong style sheet:

:root {  
  --main-bg-color: coral;  
}  
  
#div1 {  
  background-color: var(--main-bg-color);  
}  
  
#div2 {  
  background-color: var(--main-bg-color);  
}

Ví dụ dưới đây sử dụng hàm var() để chèn một vài giá trị thuộc tính tùy chỉnh:

:root {  
  --main-bg-color: coral;  
  --main-txt-color: blue;  
  --main-padding: 15px;  
}  
  
#div1 {  
  background-color: var(--main-bg-color);  
  color: var(--main-txt-color);  
  padding: var(--main-padding);  
}  
  
#div2 {  
  background-color: var(--main-bg-color);  
  color: var(--main-txt-color);  
  padding: var(--main-padding);  
}

***2.3.2.21 CSS Box sizing***

* + CSS Boxing sizing

Thuộc tính CSS box-sizing cho phép chúng ta bao gồm phần đệm và đường viền trong tổng chiều rộng và chiều cao của một phần tử.

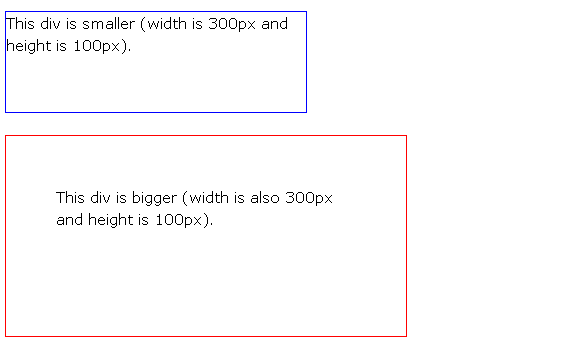
* + Without the CSS box-sizing Property

Theo mặc định, chiều rộng và chiều cao của một phần tử được tính như sau:

* + Chiều rộng + đệm + viền = chiều rộng thực tế của một phần tử
  + Chiều cao + đệm + viền = chiều cao thực tế của một phần tử

Điều này có nghĩa : khi bạn thiết lập chiều rộng / chiều cao của một phần tử, phần tử này thường xuất hiện lớn hơn kích thước bạn đã thiết lập (vì đường viền và đệm của phần tử được thêm vào của phần tử quy định chiều rộng/chiều cao).

Hình minh họa sau đâ có thấy hai phần tử <div> với cùng chiều rộng và chiều cao:



Hai phần tử <div> ở trên kết thúc với kích cỡ khác nhau trong kết quả (vì div2 đã quy định đệm):

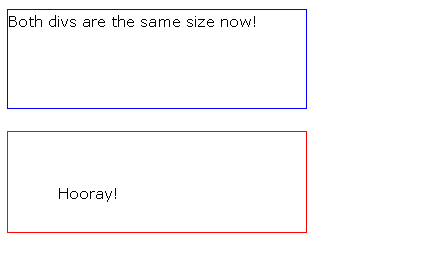
.div1 {  
  width: 300px;  
  height: 100px;  
  border: 1px solid blue;  
}  
  
.div2 {  
  width: 300px;  
  height: 100px;  
  padding: 50px;  
  border: 1px solid red;  
}

Các thuộc tính box-sizing giải quyết vấn đề này.

* + With the CSS box-sizing Property

Thuộc tính box-sizing cho phép chúng ta bao gồm phần đệm và đường viền trong tổng chiều rộng và chiều cao của một phần tử.

Nếu bạn đặt box-sizing:border-box; trên một phần tử đệm và đường viền sẽ bao gồm cả chiều rộng và chiều cao:



Đây là ví dụ giống ví dụ ở trên, với box-sizing:border-box; được thêm vào cả hai phần tử <div>:

.div1 {  
  width: 300px;  
  height: 100px;  
  border: 1px solid blue;  
  box-sizing: border-box;  
}  
  
.div2 {  
  width: 300px;  
  height: 100px;  
  padding: 50px;  
  border: 1px solid red;  
  box-sizing: border-box;  
}

Kết quả khi sử dụng box-sizing: border-box; tốt hơn rất nhiều, nhiều nhà phát triển muốn tất cả các phần tử trên trang của họ làm việc theo cách này.

Đoạn mã dưới đây đảm bảo rằng tất cả các yếu tố có kích thước theo một cách trực quan hơn . Nhiều trình duyệt đã sử dụng box-sizing: border-box; cho nhiều phần tử form( nhưng không phải tất cả - đó là lý do tại sao đầu vào và vùng văn bản trông khác nhau ở width:100%;)

Áp dụng điều này cho tất cả các phần tử là an toàn và thông minh:

\* {  
  box-sizing: border-box;  
}

***2.3.2.22 CSS Flexbox***

* + CSS Flexbox Layout Module

Trước khi có bố cục mô-đun Flexbox, đã có bốn bố cục mô-đun:

* + Block, cho các phần trong một trang web.
  + Inline, cho văn bnar
  + Table, cho dữ liệu bảng hai chiều
  + Positioned, cho vị trí cụ thể của một phần tử.

Bố cục mô-đun Flexible Box giúp dễ dàng thiết kế cấu trúc bố cục đáp ứng linh hoạt mà không cần sử dụng float hoặc positioning.

* + Các trình duyệt hỗ trợ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Chrome | Internet Explorer | Firefox | Safari | Opera |
| 29.0 | 11.0 | 22.0 | 10 | 48 |

* + Flexbox Elements

Để bắt đầu sử dụng các mô hình Flexbox, đầu tiên bạn cần phải xác định một flex container.



Phần tử bên trên đại diện cho một flex container (vùng màu xanh) với 3 mục flex.

<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div>3</div>  
</div>

* + Parent Element (Container)

Flex container trở nên linh hoạt bằng cách thiết lập thuộc tính display:

.flex-container {  
  display: flex;  
}

Các thuộc tính flex container:

* + Flex-director
  + Flex-wrap
  + Flex-flow
  + Justify-content
  + Align-items
  + Align-content
  + Thuộc tính flex-direction

Thuộc tính flex-direction xác định theo hướng mà container muốn xếp các mục flex.

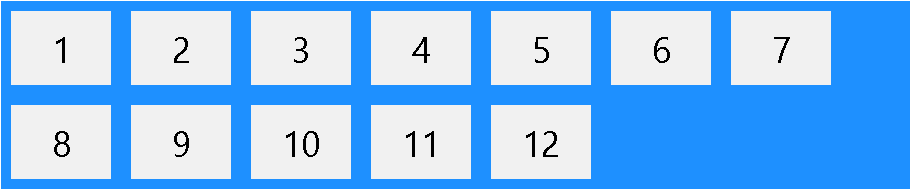


.flex-container {  
  display: flex;  
  flex-direction: column;  
}

* + Thuộc tính flex-wrap

Thuộc tính flex-wrap xác định xem các mục flex có nên được bao hay không.

Ví dụ dưới đây có 12 mục flex, để minh họa rõ hơn về thuộc tính flex-wrap:



.flex-container {  
  display: flex;  
  flex-wrap: wrap;  
}

Ví dụ:

.flex-container {  
  display: flex;  
  flex-wrap: nowrap;  
}

.flex-container {  
  display: flex;  
  flex-wrap: wrap-reverse;  
}

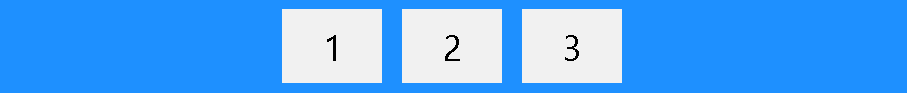
* + Thuộc tính flex-flow

Thuộc tính flex-flow là một thuộc tính viêt tắt để thiết lập cả hai thuộc tính flex-direction và flex-wrap.

.flex-container {  
  display: flex;  
  flex-flow: row wrap;  
}

* + Thuộc tính justify-content

Thuộc tính justify-content được sử dụng để sắp xếp các mục flex:



.flex-container {  
  display: flex;  
  justify-content: center;  
}

Ví dụ: Các giá trị flex-start gắn các mục flex vào đầu container

.flex-container {  
  display: flex;  
  justify-content: flex-start;  
}

Các giá trị flex-end gắn các mục flex ở phần cuối của container

.flex-container {  
  display: flex;  
  justify-content: flex-end;  
}

* + Thuộc tính align-items

Thuộc tính align-items được sử dụng để sắp xếp các mục flex theo chiều dọc



Giá trị center gắn các mục flex ở giữa container:

.flex-container {  
  display: flex;  
  height: 200px;  
  align-items: center;  
}

Ví dụ: Các flex-start gắn các mục flex ở phía trên cùng của container:

.flex-container {  
  display: flex;  
  height: 200px;  
  align-items: flex-start;  
}

Các flex-end gắn các mục flex ở phía dưới cùng của container:

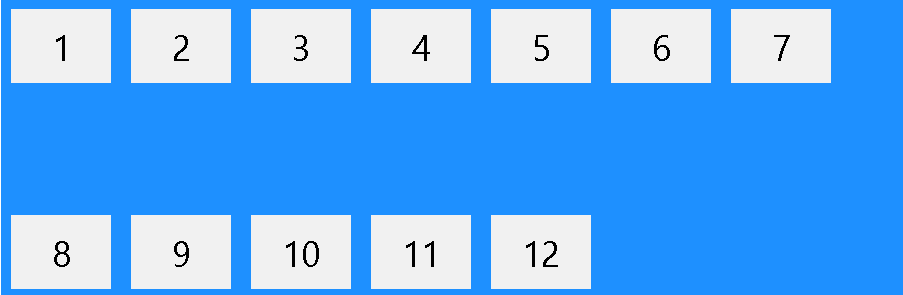
.flex-container {  
  display: flex;  
  height: 200px;  
  align-items: flex-end;  
}

Các giá trị stretch trải dài các mục flex để điền vào container:

.flex-container {  
  display: flex;  
  height: 200px;  
  align-items: stretch;  
}

* + Thuộc tính align-content

Thuộc tính align-content được sử dụng để sắp xếp các dòng flex.



Ví dụ: Giá trị space-between hiển thị các dòng flex với không gian bằng nhau giữ chúng:

.flex-container {  
  display: flex;  
  height: 600px;  
  flex-wrap: wrap;  
  align-content: space-between;  
}

Giá trị space-round hiển thị các dòng flex với không gian trước, giữa và sau chúng:

.flex-container {  
  display: flex;  
  height: 600px;  
  flex-wrap: wrap;  
  align-content: space-around;  
}

Giá trị stretch trải dài các dòng flex để đưa chúng lên không gian còn lại (mặc định):

.flex-container {  
  display: flex;  
  height: 600px;  
  flex-wrap: wrap;  
  align-content: stretch;  
}

Giá trị center hiển thị các dòng flex ở giữa của container:

.flex-container {  
  display: flex;  
  height: 600px;  
  flex-wrap: wrap;  
  align-content: center;  
}

Giá trị flex-start hiển thị các dòng flex ở đầu container:

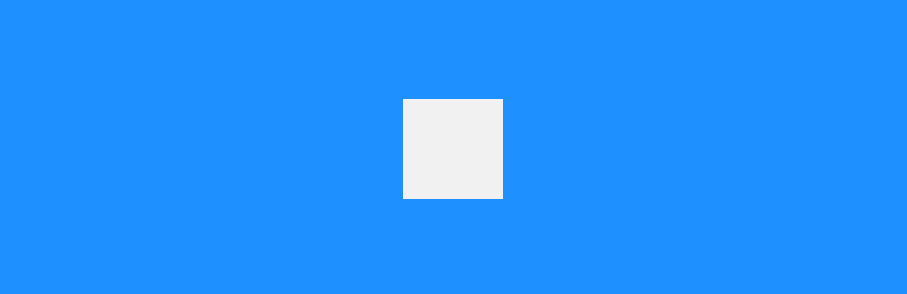
.flex-container {  
  display: flex;  
  height: 600px;  
  flex-wrap: wrap;  
  align-content: flex-start;  
}

Giá trị flex-end hiển thị các dòng flex ở cuối container:

.flex-container {  
  display: flex;  
  height: 600px;  
  flex-wrap: wrap;  
  align-content: flex-end;  
}

* + Perfect Centering

Trong ví dụ sau, chúng ta sẽ giải quyết một vấn đề định dạng phổ biến : định tâm hoàn hảo.

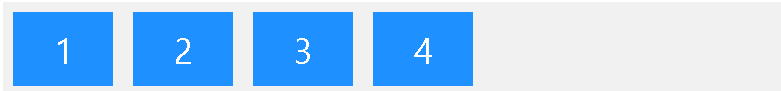


Giải pháp: đặt cả thuộc tính justify-content và align-items cho center, và mục flex sẽ trở lên định tâm hoàn hảo.

.flex-container {  
  display: flex;  
  height: 300px;  
  justify-content**: center;**  align-items**: center;**}

* + Child Elements (Items)

Các phần tử con trực tiếp của một flex container sẽ tự động trở thành các mục linh hoạt (flex).



Phần tử ở trên đại diện cho 4 mục flex màu xanh bên trong một container màu xám.

<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div>3</div>  
  <div>4</div>  
</div>

Các thuộc tính mục flex:

* + Order
  + Flex-grow
  + Flex-shrink
  + Flex-basis
  + Flex
  + Align-self
  + Thuộc tính order

Thuộc tính order xác định thứ tự của các mục flex.



Mục đầu tiên trong code không nhất thiết phải xuất hiện như mục đầu tiên trong cách bố trí.

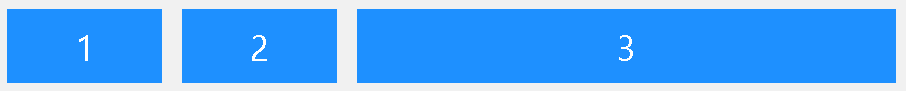
Giá trị order phải là một số, giá trị mặc định là 0.

Thuộc tính order có thể thay đổi thứ tự của các mục flex:

<div class="flex-container">  
  <div style="order: 3">1</div>  
  <div style="order: 2">2</div>  
  <div style="order: 4">3</div>  
  <div style="order: 1">4</div>  
</div>

* + Thuộc tính flex-grow

Thuộc tính flex-grow xác định số mục flex sẽ phát triển tương đối với phần còn lại của các mục flex.



Giá trị phải là số, mặc định là 0.

<div class="flex-container">  
  <div style="flex-grow: 1">1</div>  
  <div style="flex-grow: 1">2</div>  
  <div style="flex-grow: 8">3</div>  
</div>

* + Thuộc tính flex-shrink

Thuộc tính flex-shrink xác định số mục flex sẽ co lại so với phần tử còn lại của các mục flex



Giá trị phải là số, mặc định là 1.

<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div style="flex-shrink: 0">3</div>  
  <div>4</div>  
  <div>5</div>  
  <div>6</div>  
  <div>7</div>  
  <div>8</div>  
  <div>9</div>  
  <div>10</div>  
</div>

* Thuộc tính flex-basic

Thuộc tính flex-basic xác định độ dài ban đầu của một mục flex.



<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div style="flex-basis: 200px">3</div>  
  <div>4</div>  
</div>

* + Thuộc tính flex

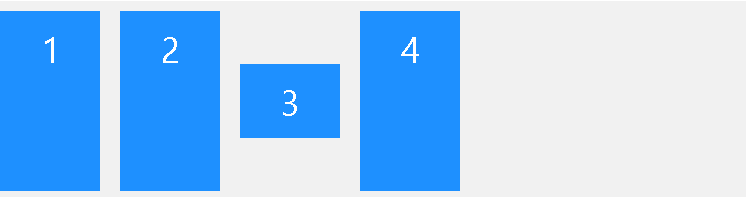
Thuộc tính flex là một thuộc tính viết tắt cho thuộc tính flex-grow, flex-shrink và flex-basis.

<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div style="flex: 0 0 200px">3</div>  
  <div>4</div>  
</div>

* + Thuộc tính align-self

Thuộc tính align-self chỉ định căn chỉnh cho mục được chọn bên trong flex container.

Thuộc tính align-self ghi đè căn chỉnh mặc định được đặt bởi thuộc tính align-items.



<div class="flex-container">  
  <div>1</div>  
  <div>2</div>  
  <div style="align-self: center">3</div>  
  <div>4</div>  
</div>

Ví dụ: căn chỉnh 3 mục flex ở giữa container:

<div class="flex-container">  
  <div>1</div>  
  <div style="align-self: flex-start">2</div>  
  <div style="align-self: flex-end">3</div>  
  <div>4</div>  
</div>

***2.3.2.23 CSS Media Queries***

* + CSS2 introduced Media Types

Quy tắc @media được giới thiệu trong CSS2, cho phép xác định các quy tắc định dạng khác nhau cho các loại phương tiện khác nhau.

Ví dụ: bạn có thể có một bộ quy tắc định dạng kiểu cho màn hình msy tính, cho một máy in, cho một thiết bị cầm tay, cho một thiết bị kiểu truyền hình,….

Thật không may, các loại phương tiện này không bao giờ nhận được nhiều sự hỗ trợ của các thiết bị, ngoại trừ phương tiện in.

* + CSS3 introduced Media Queries

Các truy vấn phương tiện trong CSS3 kế thừa ý tưởng của CSS2 media type : thay vì tìm kiếm một loại thiết bị, họ xem xét khả năng của thiết bị.

Media queries có thể được sử dụng để kiểm tra nhiều thứ, chẳng hạn như :

* + Chiều rộng và chiều cao của khung hình
  + Chiều rộng và chiều cao của thiết bị định hướng( là máy tính bảng/điện thoại ở chế độ ngang hay dọc?)
  + Giải pháp

Sử dụng truy vấn phương tiện là một kỹ thuật phổ biến để cung cấp biểu định kiểu phù hợp cho máy tính để bàn, máy tính bảng và điện thoại di động ( như điện thoại iPhone và Android)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Property | Chrome | Internet Explorer | Firefox | Safari | Opera |
| @media | 21.0 | 9.0 | 3.5 | 4.0 | 9.0 |

* + Trình duyệt hỗ trợ
  + Media Query syntax

Một truy vấn phương tiện truyền thông bao gồm một loại phương tiện truyền thông và có thể chứa một haowcj nhiều biểu thức mà giả quyết là đúng hoặc sai.

@media not|only *mediatype*and(*expressions*) { *CSS-Code;*}

Kết quả của truy vấn là đúng nếu loại phương tiện được chỉ định phù hợp với loại thiết bị mà tài liệu đang được hiển thị và tất cả các biểu thức trong truy vấn phuoeng tiện đều đúng. Khi truy vấn phương tiện là đúng, quy tắc biểu định kiểu hoặc kiểu tương ứng được áp dụng, tuân theo quy tắc xếp tầng thông thường.

Trừ khi bạn sử dụng các toán tử không hoặc duy nhất, loại phương tiện là tùy chọn và tất cả các loại sẽ được ngụ ý.

Bạn cũng có thể có các bảng định kiểu khác nhau cho các phương tiện khác nhau:

<link rel="stylesheet" media="mediatype and|not|only (expressions)" href="print.css">

|  |  |  |
| --- | --- | --- |
| **Giá trị** | | **Mô tả** |
| all | Sử dụng cho tất cả các loại thiết bị phương tiện truyền thông | |
| print | Sử dụng cho máy in | | |
| screen | Sử dụng cho màn hình máy tính, máy tính bảng, điện thoại thông minh… | | |
| speech | Được sử dụng cho các trình đọc màn hình “read” trang thành tiếng | | |

* + Media Queries Simple Examples

@media screen and (min-width: 480px) {  
  body {  
    background-color: lightgreen;  
  }  
}

@media screen and (min-width: 480px) {  
  #leftsidebar {width: 200px; float: left;}  
  #main {margin-left: 216px;}  
}

***2.3.2.24 CSS MQ Examples***

* + CSS Media Queries – more examples

Chúng ta có thể thay đổi màu nền cho các thiết bị khác nhau:

/\* Set the background color of body to tan \*/  
body {  
  background-color: tan;  
}  
  
/\* On screens that are 992px or less, set the background color to blue \*/  
@media screen and (max-width: 992px) {  
  body {  
    background-color: blue;  
  }  
}  
  
/\* On screens that are 600px or less, set the background color to olive \*/  
@media screen and (max-width: 600px) {  
  body {  
    background-color: olive;  
  }  
}

* + Media Queries for menu

/\* The navbar container \*/  
.topnav {  
  overflow: hidden;  
  background-color: #333;  
}  
  
/\* Navbar links \*/  
.topnav a {  
  float: left;  
  display: block;  
  color: white;  
  text-align: center;  
  padding: 14px 16px;  
  text-decoration: none;  
}  
  
/\* On screens that are 600px wide or less, make the menu links stack on top of each other instead of next to each other \*/  
@media screen and (max-width: 600px) {  
  .topnav a {  
    float: none;  
    width: 100%;  
  }  
}

* + Media Queries for Colums

Một sử dụng phổ biến của các truy vấn phương tiện truyền thông là để tạo một bố cục linh hoạt.

Ví dụ:

/\* Create four equal columns that floats next to each other \*/  
.column {  
  float: left;  
  width: 25%;  
}  
  
/\* On screens that are 992px wide or less, go from four columns to two columns \*/  
@media screen and (max-width: 992px) {  
  .column {  
    width: 50%;  
  }  
}  
  
/\* On screens that are 600px wide or less, make the columns stack on top of each other instead of next to each other \*/  
@media screen and (max-width: 600px) {  
  .column {  
    width: 100%;  
  }  
}

* + Hide Element with Media Queries

/\* If the screen size is 600px wide or less, hide the element \*/  
@media screen and (max-width: 600px) {  
  div.example {  
    display: none;  
  }  
}

* + Change Font size with Media Queries

/\* If screen size is more than 600px wide, set the font-size of <div> to 80px \*/  
@media screen and (min-width: 600px) {  
  div.example {  
    font-size: 80px;  
  }  
}  
  
/\* If screen size is 600px wide, or less, set the font-size of <div> to 30px \*/  
@media screen and (max-width: 600px) {  
  div.example {  
    font-size: 30px;  
  }  
}

* + Flexiable Image Gallery
  + Flexible Website
  + Orientation: Portrait/Landscape

Truy vấn phương tiện truyền thông cũng có thể được sử dụng để thay đổi bố cục của một trang tùy thuộc vào định hướng của trình duyệt.

@media only screen and (orientation: landscape) {  
  body {  
    background-color: lightblue;  
  }  
}

* + Min Width to Max Width

Bạn có thể sử dụng giá trị (max-width:..) và (min-width:..) để thiết lập chiều rộng nhỏ nhất và lớn nhất.

@media screen and (max-width: 900px) and (min-width: 600px) {  
  div.example {  
    font-size: 50px;  
    padding: 50px;  
    border: 8px solid black;  
    background: yellow;  
  }  
}

Sử dụng một giá trị gia tăng: trong ví dụ dưới đây, chúng ta thêm một truy vấn phương tiện truyền thông vào một truy vấn hiện có bằng cách sử dụng dấu phẩy (điều này sẽ hoạt động giống như một toán tử OR):

/\* When the width is between 600px and 900px OR above 1100px - change the appearance of <div> \*/  
@media screen and (max-width: 900px) and (min-width: 600px), (min-width: 1100px) {  
  div.example {  
    font-size: 50px;  
    padding: 50px;  
    border: 8px solid black;  
    background: yellow;  
  }  
}

**2.3.3 Responsive**

***2.3.3.1. RWD INTRO***

1. Thiết kế Web đáp ứng là gì?

Thiết kế web đáp ứng làm cho trang web của bạn trông đẹp trên tất cả các thiết bị.

2. Thiết kế web đáp ứng chỉ sử dụng HTML và CSS.

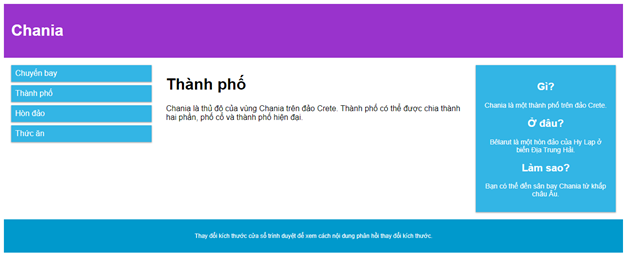
Thiết kế web đáp ứng không phải là một chương trình hoặc JavaScript.

Thiết kế cho trải nghiệm tốt nhất cho tất cả người dùng

Các trang web có thể được xem bằng nhiều thiết bị khác nhau: máy tính để bàn, máy tính bảng và điện thoại. Trang web của bạn sẽ trông tốt và dễ sử dụng, bất kể thiết bị.

Nó được gọi là thiết kế web đáp ứng khi bạn sử dụng CSS và HTML để thay đổi kích thước, ẩn, thu nhỏ, phóng to hoặc di chuyển nội dung để làm cho nó trông đẹp hơn trên bất kỳ màn hình nào

Các trang web không nên bỏ thông tin để phù hợp với các thiết bị nhỏ hơn mà nên điều chỉnh nội dung của nó để phù hợp với bất kỳ thiết bị nào

**

***2.3.3.2. RWD viewport***

*1.*Khung nhìn là gì?

Khung nhìn là khu vực hiển thị của người dùng trên trang web.

Chế độ xem thay đổi theo thiết bị và sẽ nhỏ hơn trên điện thoại di động so với trên màn hình máy tính.

Trước máy tính bảng và điện thoại di động, các trang web chỉ được thiết kế cho màn hình máy tính và thông thường các trang web có thiết kế tĩnh và kích thước cố định.

Sau đó, khi chúng tôi bắt đầu lướt internet bằng máy tính bảng và điện thoại di động, các trang web có kích thước cố định quá lớn để phù hợp với chế độ xem. Để khắc phục điều này, các trình duyệt trên các thiết bị đó thu nhỏ toàn bộ trang web để vừa với màn hình.

Điều này không hoàn hảo !! Nhưng một sửa chữa nhanh chóng.

2. Đặt Chế độ xem

HTML5 đã giới thiệu một phương pháp để cho phép các nhà thiết kế web kiểm soát chế độ xem, thông qua <meta>thẻ.

Bạn nên bao gồm <meta>yếu tố chế độ xem sau trong tất cả các trang web của mình:

<meta name="viewport" content="width=device-width, initial-scale=1.0">

Phần <meta>tử khung nhìn cung cấp cho hướng dẫn trình duyệt về cách kiểm soát kích thước và tỷ lệ của trang.

Phần width=device-widthđặt chiều rộng của trang để theo chiều rộng màn hình của thiết bị (sẽ thay đổi tùy theo thiết bị).

Phần initial-scale=1.0đặt mức thu phóng ban đầu khi trình duyệt được tải lần đầu tiên.

3.Kích thước nội dung cho Viewport

Người dùng được sử dụng để cuộn các trang web theo chiều dọc trên cả máy tính để bàn và thiết bị di động - nhưng không phải theo chiều ngang!

Vì vậy, nếu người dùng buộc phải cuộn theo chiều ngang hoặc thu nhỏ, để xem toàn bộ trang web, điều đó dẫn đến trải nghiệm người dùng kém.

Một số quy tắc bổ sung cần tuân theo:

- KHÔNG sử dụng các yếu tố chiều rộng cố định lớn - Ví dụ: nếu hình ảnh được hiển thị ở chiều rộng rộng hơn khung nhìn, nó có thể khiến chế độ xem cuộn theo chiều ngang. Hãy nhớ điều chỉnh nội dung này cho vừa với chiều rộng của khung nhìn.

- KHÔNG để nội dung dựa vào chiều rộng của chế độ xem cụ thể để hiển thị tốt - Vì kích thước và chiều rộng màn hình trong pixe pixel rất khác nhau giữa các thiết bị, nội dung không nên dựa vào chiều rộng của chế độ xem cụ thể để hiển thị tốt.

-Sử dụng truy vấn phương tiện CSS để áp dụng kiểu dáng khác nhau cho màn hình nhỏ và lớn - Đặt độ rộng CSS tuyệt đối lớn cho các thành phần trang sẽ khiến phần tử quá rộng cho chế độ xem trên thiết bị nhỏ hơn. Thay vào đó, hãy xem xét sử dụng các giá trị chiều rộng tương đối, chẳng hạn như chiều rộng: 100%. Ngoài ra, hãy cẩn thận khi sử dụng các giá trị định vị tuyệt đối lớn. Nó có thể khiến phần tử rơi ra ngoài chế độ xem trên các thiết bị nhỏ*.*

***2.3.3.3. RWD Grid view***

1. Chế độ xem lưới là gì?

Nhiều trang web dựa trên chế độ xem dạng lưới, có nghĩa là trang được chia thành các cột



Sử dụng chế độ xem lưới rất hữu ích khi thiết kế trang web. Nó làm cho nó dễ dàng hơn để đặt các yếu tố trên trang



Chế độ xem lưới phản hồi thường có 12 cột và có tổng chiều rộng 100% và sẽ thu nhỏ và mở rộng khi bạn thay đổi kích thước cửa sổ trình duyệt.

2. Xây dựng chế độ xem lưới đáp ứng

Hãy bắt đầu xây dựng chế độ xem lưới đáp ứng.

Trước tiên, đảm bảo rằng tất cả các thành phần HTML có thuộc box-sizing tính được đặt thành border-box. Điều này đảm bảo rằng phần đệm và đường viền được bao gồm trong tổng chiều rộng và chiều cao của các phần tử.

Thêm mã sau vào CSS của bạn:

\* {  
  box-sizing: border-box;  
}

***2.3.3.4 RWD Media queries***

Media CSS là gì?

Media CSS là một "công nghệ" được giới thiệu trong Css3. Nguyên tắc nó sử dụng trên thông qua các thông số kích thước màn hình được khai báo thông qua @media Với công cụ này chúng ta có thể phân đoạn Css chúng ta ra nhiều phần khác nhau tương ứng với kích thước của các loại thiết bị. Thuộc tính này có cú pháp như sau:

@media not|only mediatype and (media feature) {

CSS-Code;

}

Trong đó mediatype gồm các thuộc tính hay sử dụng sau:

• all: Dùng cho mọi thiết bị

• print: Dùng cho máy in

• screen: Dùng cho máy tính và các thiết bị smart phone

Thực tế vẫn còn nhiều nữa nhưng với lập trình web thì chúng ta thường sử dụng ba thuộc tính đó thôi. Và trước khi đi vào tìm hiểu các thuộc tính thì ban phải phân biệt hai khái niệm sau:

• Device: Là thiết bị sử dụng website như Laptop, Desktop, Iphone, ..

• Viewport: Là kích thước hiển thị của giao diện.

<meta name="viewport" content="width=device-width, initial-scale=1">

Và media featured thì gồm các thuộc tính sau:

• aspect-ratio: Tỉ lệ giữa chiều rộng và chiều cao của viewport

• min-aspect-ratio: Tỉ lệ tối thiểu giữa chiều rộng và chiều cao của viewport

• max-aspect-ratio: Tỉ lệ tôi đa giữa chiều rộng và chiều cao của viewport

• color: Số bits cho mỗi màu sắc của device

• color-index: Số lượng màu sắc mà device có thể hiển thị

• device-aspect-ratio: Tỉ lệ giữa chiều rộng và chiều cao của device

• max-device-aspect-ratio: Tỉ lệ tối đa giữa chiều rộng và chiều cao của device

• min-device-aspect-ratio: Tỉ lệ tối thiểu giữa chiều rộng và chiều cao của device

• device-height: Chiều cao của device

• device-width: Chiều rộng của device

• height: Chiều cao của viewport

• width: Chiều rộng của viewport

• max-width: Chiều rộng tối đa của viewport

• min-width: Chiều rộng tối thiểu của viewport

• max-height: Chiều cao tối đa của viewport

• min-height: Chiều cao tối thiểu của viewport

• min-device-width: Chiều rộng tối thiểu của device

• max-device-width: Chiều rộng tối đa của device

• min-device-height: Chiều cao tối thiểu của device

• max-device-height: Chiều cao tối đa của device

• orientation: Định hướng của khung nhìn (xoay hoặc không xoay thiết bị)

• resolution: Độ phân giải của thiết bị đầu ra (sử dụng dpi hoặc dpcm)

Ví dụ: Nếu chiều rộng của trình duyệt nhỏ hơn hoặc bằng 480px thì ẩn class .large.

@media only screen and (max-width: 480px){

.large{

display: none;

}

}

2. Sử dụng @media để tạo Responsive Website

Ngoài max-width chúng ta còn có tham số hay được sử dụng như min-width. Vậy việc sử dụng Media query nhằm mục đích đưa ra các phân đoạn để viết Css theo độ rộng màn hình các thiết bị. Vậy bản chất của RESPONSIVE là sự kết hợp của media query và tính ưu tiên thực thi của Css trên cùng một đối tượng.

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<title>**Media responsive**</title>**
5. **<meta** charset="UTF-8"**>**
6. **<meta** name="viewport" content="width=device-width, initial-scale=1.0"**>**
7. **<style>**
8. .left, .right{
9. height: 400px;
10. }
11. .left{
12. background: blue;
13. float: left;
14. width: 60%;
15. }
16. .right{
17. background: red;
18. float: right;
19. width: 40%;
20. }
22. **</style>**
23. **</head>**
24. **<body>**
25. **<div** class="container"**>**
26. **<div** class="left"**>**
27. LEFT
28. **</div>**
29. **<div** class="right"**>**
30. RIGHT
31. **</div>**
32. **</div>**
33. **</body>**
34. **</html>**

• Nếu kích thước trình duyệt nhỏ hơn 769px thì hiển thị 1 hàng dọc

• Ngược lại thì hiển thị hai hàng dọc

Lúc này ta sẽ thiết lập CSS như sau:

1. @media only screen and (**max-width**: 768px){
2. .left, .right{
3. **float**: none;
4. **width**: 100%;
5. }
6. }

3. Sử dụng @media với các thiết bị di động và máy tính

Với danh sách các thuộc tính của @media thì ta dễ dàng phát hiện ra các thiết bị. Và sau đây là danh sách các câu query @media cho các thiết bị thông dụng.

PC first trong Rseponsive

PC first là khái niệm để chỉ tuần tự responsive giao diện từ màn hình to xuống màn hình nhỏ

Để làm việc với mô hình này chúng ta sử dụng max-width trong media query

Dưới đây là các media query điển hình mà ta cần thêm vào dự án.

1. /\*Ipad ngang(1024 x 768)\*/
2. @media screen and (**max-width**: 1024px){
3. }
4. /\*Ipad dọc(768 x 1024)\*/
5. @media screen and (**max-width**: 768px){
7. }
8. /\*Tablet nhỏ(480 x 640)\*/
9. @media screen and (**max-width**: 480px){
10. }
11. /\*Iphone(480 x 640)\*/
12. @media screen and (**max-width**: 320px){
13. }
14. /\*Smart phone nhỏ\*/
15. @media screen and (**max-width**: 240px){

Ngoài ra, nếu mình cần responsive trên nhiều thiết bị hơn thì công việc của chúng ta là cần xác định kích thước của nó và thêm vào danh sách media query kia theo thứ tự màn hình to ở trên màn hình nhỏ query ở dưới. Theo cách này thì khi một Selector trong css cần style đi qua từ màn hình to đến nhỏ sẽ được thay đổi theo thứ tự ưu tiên Trái ngược với PC First chúng ta cùng đi qua Mobile First

Mobile First trong Responsive

Tuần tự tiến trình responsive của chúng ta xuất phát từ màn hình nhỏ và xây dựng dần lên thiết bị có kích thước to

Để làm việc với mô hình này chúng ta sử dụng tham số min-width trong media query.

/Với mô hình này thì khi style Css cho cùng một đối tượng thì theo thứ tự được ưu tiên từ màn hình nhỏ đến to, càng về sau các các Css ở các query màn hình to hơn được ưu tiên hơn.

***2.3.3.5 RWD Images***

**Sử dụng thuộc tính chiều rộng**

Nếu thuộc widthtính được đặt thành tỷ lệ phần trăm và chiều cao được đặt thành "tự động", hình ảnh sẽ được phản hồi và tăng tỷ lệ lên xuống:

Thí dụ

img {

width: 100%;

height: auto;

}

**Sử dụng Thuộc tính chiều rộng tối đa**

Nếu thuộc max-widthtính được đặt thành 100%, hình ảnh sẽ thu nhỏ lại nếu phải, nhưng không bao giờ tăng tỷ lệ lớn hơn kích thước ban đầu của nó:

Thí dụ

img {

max-width: 100%;

height: auto;

}

Thêm một hình ảnh vào trang web ví dụ

Thí dụ

img {

width: 100%;

height: auto;

}

**Hinh nên**

Hình nền cũng có thể đáp ứng với thay đổi kích thước và tỷ lệ.

Ở đây chúng tôi sẽ chỉ ra ba phương pháp khác nhau:

1. Nếu thuộc background-sizetính được đặt thành "chứa", hình nền sẽ chia tỷ lệ và cố gắng khớp với vùng nội dung. Tuy nhiên, hình ảnh sẽ giữ tỷ lệ khung hình của nó (mối quan hệ tỷ lệ giữa chiều rộng và chiều cao của hình ảnh):

Thí dụ

1. div {
2. **width**: 100%;
3. **height**: 400px;
4. **background-image**: url('img\_flowers.jpg');
5. **background-repeat**: no-repeat;
6. background-**size**: contain;
7. **border**: 1px solid red;
8. }

2. Nếu thuộc background-sizetính được đặt thành "100% 100%", hình nền sẽ kéo dài để bao phủ toàn bộ khu vực nội dung:

Thí dụ

1. div {
2. **width**: 100%;
3. **height**: 400px;
4. **background-image**: url('img\_flowers.jpg');
5. background-**size**: 100% 100%;
6. **border**: 1px solid red;
7. }

3. Nếu thuộc background-sizetính được đặt thành "che", hình nền sẽ chia tỷ lệ để bao phủ toàn bộ khu vực nội dung. Lưu ý rằng giá trị "che phủ" giữ tỷ lệ khung hình và một số phần của hình nền có thể được cắt bớt:

Thí dụ

1. div {
2. **width**: 100%;
3. **height**: 400px;
4. **background-image**: url('img\_flowers.jpg');
5. background-**size**: cover;
6. **border**: 1px solid red;
7. }

**Hình ảnh khác nhau cho các thiết bị khác nhau**

Một hình ảnh lớn có thể hoàn hảo trên màn hình máy tính lớn, nhưng vô dụng trên một thiết bị nhỏ. Tại sao tải một hình ảnh lớn khi bạn phải thu nhỏ nó xuống? Để giảm tải hoặc vì bất kỳ lý do nào khác, bạn có thể sử dụng truy vấn phương tiện để hiển thị các hình ảnh khác nhau trên các thiết bị khác nhau.

Đây là một hình ảnh lớn và một hình ảnh nhỏ hơn sẽ được hiển thị trên các thiết bị khác nhau:

Thí dụ

1. /\* For width smaller than 400px: \*/
2. body {
3. **background-image**: url('img\_smallflower.jpg');
4. }
6. /\* For width 400px and larger: \*/
7. @media only screen and (**min-width**: 400px) {
8. body {
9. **background-image**: url('img\_flowers.jpg');
10. }
11. }

Bạn có thể sử dụng truy vấn phương tiện min-device-width, thay vì min-widthkiểm tra độ rộng của thiết bị, thay vì chiều rộng của trình duyệt. Sau đó, hình ảnh sẽ không thay đổi khi bạn thay đổi kích thước cửa sổ trình duyệt:

Thí dụ

/\* For devices smaller than 400px: \*/

**body**{

**background-image**:url('img\_smallflower.jpg');

}

/\* For devices 400px and larger: \*/

@mediaonly screen and (min-device-width: 400px) {

**body**{

**background-image**:url('img\_flowers.jpg');

}

}

***2.3.3.6 RWD Videos***

**Sử dụng thuộc tính chiều rộng**

Nếu thuộc width tính được đặt thành 100%, trình phát video sẽ phản hồi và mở rộng quy mô:

Thí dụ

video {

width: 100%;

height: auto;

}

Lưu ý rằng trong ví dụ trên, trình phát video có thể được thu nhỏ lại để lớn hơn kích thước ban đầu của nó. Một giải pháp tốt hơn, trong nhiều trường hợp, sẽ là sử dụng max-width thay thế.

**Sử dụng Thuộc tính chiều rộng tối đa**

Nếu max-widthđược đặt thành 100%, trình phát video sẽ giảm quy mô nếu có, nhưng không bao giờ mở rộng quy mô lớn hơn kích thước ban đầu của nó:

Thí dụ

video {

max-width: 100%;

height: auto;

}

***2.3.3.7 RWD Framework***

Có rất nhiều khung CSS hiện có cung cấp thiết kế đáp ứng.

Chúng miễn phía và dễ sử dụng.

* + **Sử dụng W3.CSS**

Một cách tuyệt vời để tạo ra một thiết kế đáp ứng là sử dụng một stylesheet đáp ứng như W3.CSS.

W3.CSS làm cho nó dễ dàng để phát triển các trang mà trông đẹp ở bất kỳ kích thước : máy tính để bàn, máy tính xách tay, máy tính bảng hay điện thoại.

Ví dụ:

<!DOCTYPE html>  
<html>  
<meta name="viewport" content="width=device-width, initial-scale=1">  
<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">  
<body>  
  
<div class="w3-container w3-green">  
  <h1>W3Schools Demo</h1>  
  <p>Resize this responsive page!</p>  
</div>  
  
<div class="w3-row-padding">  
  <div class="w3-third">  
    <h2>London</h2>  
    <p>London is the capital city of England.</p>  
    <p>It is the most populous city in the United Kingdom,  
    with a metropolitan area of over 13 million inhabitants.</p>  
  </div>  
  
  <div class="w3-third">  
    <h2>Paris</h2>  
    <p>Paris is the capital of France.</p>  
    <p>The Paris area is one of the largest population centers in Europe,  
    with more than 12 million inhabitants.</p>  
  </div>  
  
  <div class="w3-third">  
    <h2>Tokyo</h2>  
    <p>Tokyo is the capital of Japan.</p>  
    <p>It is the center of the Greater Tokyo Area,  
    and the most populous metropolitan area in the world.</p>  
  </div>  
</div>  
  
</body>  
</html>

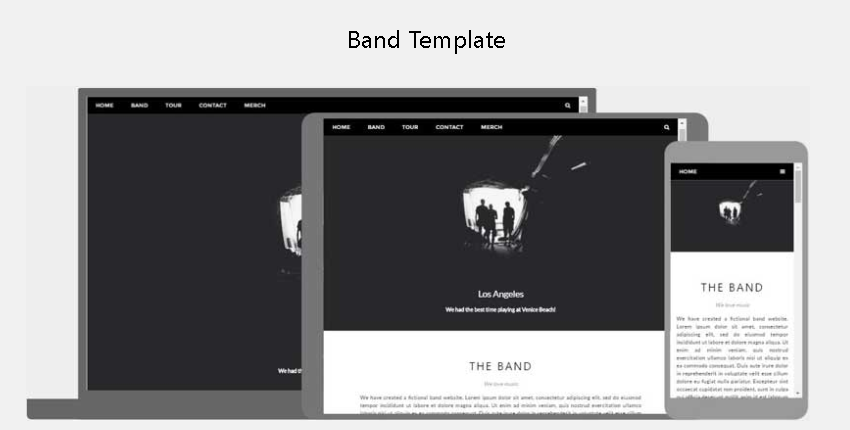
* + **Bootstrap**

Một khuôn khổ phổ biến là Bootstrap, nó sử dụng HTML, CSS và jQuery để tạo một trang web đáp ứng.

<!DOCTYPE html>  
<html lang="en">  
<head>  
<title>Bootstrap Example</title>  
<meta charset="utf-8">  
<meta name="viewport" content="width=device-width, initial-scale=1">  
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">  
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>  
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script>  
</head>  
<body>  
  
<div class="container">  
  <div class="jumbotron">  
    <h1>My First Bootstrap Page</h1>  
  </div>  
  <div class="row">  
    <div class="col-sm-4">  
      ...  
    </div>  
    <div class="col-sm-4">  
      ...  
    </div>  
    <div class="col-sm-4">  
    ...  
    </div>  
  </div>  
</div>  
  
</body>  
</html>

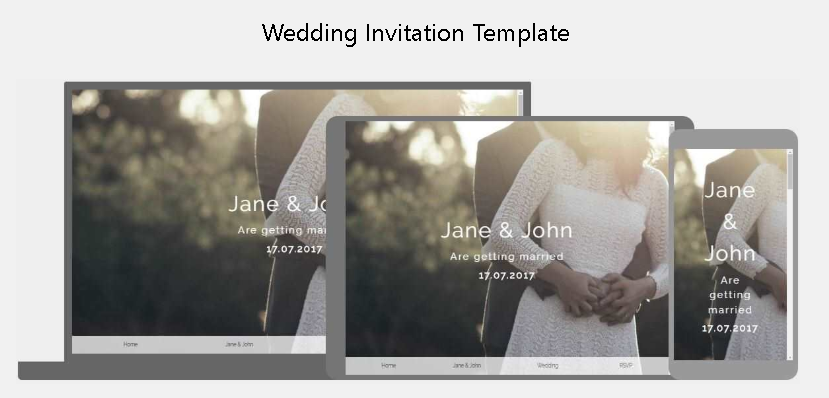
**2.3.3.8 RWD Templates**

* + W3.CSS Wensite templates









Xem thêm tại W3.CSS.

**2.3.4 Gird**

1. Bố cục lưới

Mô-đun bố trí lưới CSS cung cấp một hệ thống bố cục dựa trên lưới, với các hàng và cột, giúp thiết kế trang web dễ dàng hơn mà không phải sử dụng phao và định vị.

1. Hỗ trợ trình duyệt

Các thuộc tính lưới được hỗ trợ trong tất cả các trình duyệt hiện đại.

1. Các yếu tố lưới

Bố cục lưới bao gồm một phần tử cha, với một hoặc nhiều phần tử con.

<div class="grid-container">

<div class="grid-item">1</div>

<div class="grid-item">2</div>

<div class="grid-item">3</div>

<div class="grid-item">4</div>

<div class="grid-item">5</div>

<div class="grid-item">6</div>

<div class="grid-item">7</div>

<div class="grid-item">8</div>

<div class="grid-item">9</div>

</div>

1. Hiển thị tài sản

Một phần tử HTML trở thành một thùng chứa lưới khi thuộc tính hiển thị của nó được đặt thành lưới hoặc lưới nội tuyến.

.grid-container {

display: grid;

}

.grid-container {

display: inline-grid;

}

Tất cả con trực tiếp của container lưới sẽ tự động trở thành các mục lưới

1. Cột lưới

Các đường thẳng đứng của các mục lưới được gọi là cột.

1. Hàng lưới

Các đường ngang của các mục lưới được gọi là hàng.

1. Khoảng cách lưới

Khoảng cách giữa mỗi cột / hàng được gọi là khoảng trống.

Bạn có thể điều chỉnh kích thước khoảng cách bằng cách sử dụng một trong các thuộc tính sau:

lưới-cột-khoảng cách

lưới-hàng-khoảng cách

khoảng cách lưới

Thí dụ

Thuộc tính lưới-cột-khoảng cách đặt khoảng cách giữa các cột:

.grid-container {

display: grid;

grid-column-gap: 50px;

}

1. Đường lưới

Các dòng giữa các cột được gọi là dòng cột.

Các dòng giữa các hàng được gọi là dòng hàng.

***2.3.4.2 Grid Container***

Tạo một grid container bằng cách thiết lập thuộc tính display với giá trị grid hoặc inline-grid khi đó tất cả các item bên trong grid sẽ trở thành grid item.

***2.3.4.3 Grid Item***

* Child Elements (Items)

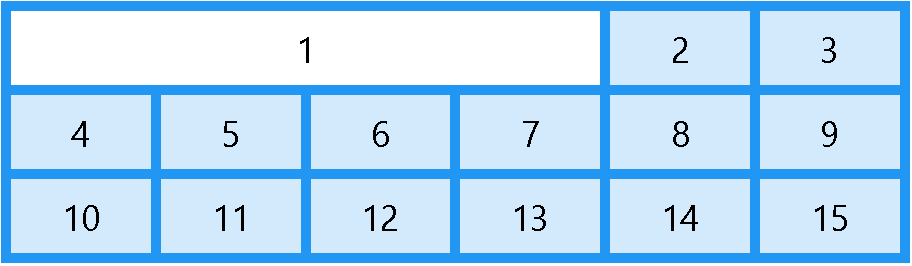
Một grid container bao gồm các mục grid.

Mặc định, một container có một mục grid cho mỗi cột, trong mỗi hàng, nhưng bạn có thể định kiểu các mục grid để chúng trải dài trên nhiều cột và/hoặc hàng.

* Thuộc tính grid-column

Thuộc tính frid-column xác định các cột để đặt mục.

Bạn có thể xác định nơi các mục sẽ bắt đầu và nơi các mục sẽ kết thúc.



Chú ý: thuộc tính grid-column là thuộc tính viết tắt của thuộc tính grid-column-start và grid-column-end.

Để đặt một mục, bạn có thẻ tham khảo số dòng, hoặc sử dụng từ khóa “span” để xác định số lượng mục sẽ trải dài.

Ví dụ: tạo “item1” bắt đầu ở cột 1 và kết thúc ở cột 5.

.item1 {  
  grid-column: 1 / 5;  
}

Tạo “item1” bắt đầu ở cột 1 và trải dài 3 cột:

.item1 {  
  grid-column: 1 / span 3;  
}

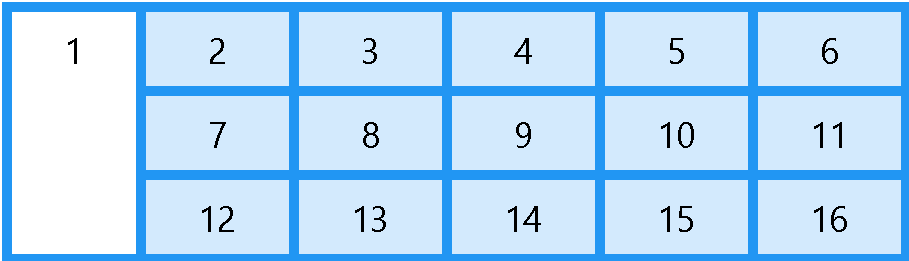
Tạo “item2” bắt đầu ở cột 2 và trải dài 3 cột:

.item2 {  
  grid-column: 2 / span 3;  
}

* Thuộc tính grid-row

Thuộc tính grid-row xác định hàng nào sẽ đặt một mục.

Bạn có thể định nghĩa nơi mục bắt đầu và nơi mục sẽ kết thúc.



Chú ý: thuộc tính grid-row là thuộc tính viết tắt của thuộc tính grid-row-start và grid-row-end.

Để đặt một mục, bạn có thẻ tham khảo số dòng, hoặc sử dụng từ khóa “span” để xác định số lượng mục sẽ trải dài.

Ví dụ: tạo “item1” bắt đầu ở row-line 1 và kết thúc ở row-line 4:

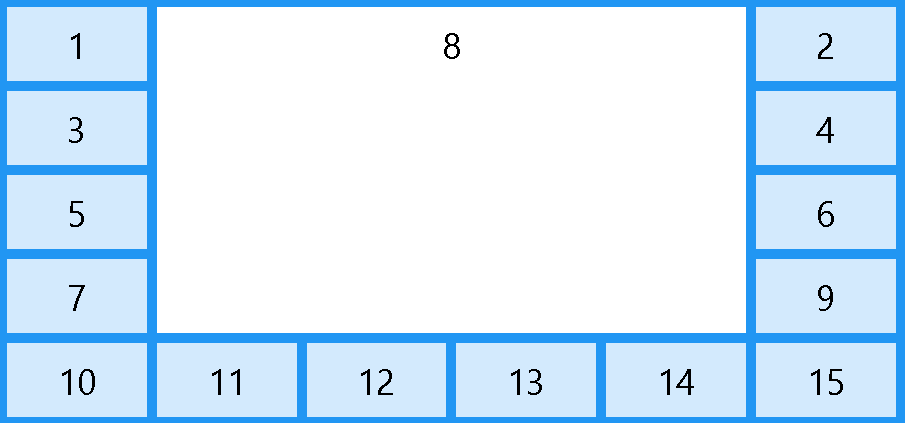
.item1 {  
  grid-row: 1 / 4;  
}

Tạo “item1” bắt đầu ở hàng 1 và trải dài 2 hàng:

.item1 {  
  grid-row: 1 / span 2;  
}

* Thuộc tính grid-area

Thuộc tính grid-are có thể được sử dụng như một thuộc tính viết tắt cho các thuộc tính grid-row-start, grid-row-end và grid-column-end.



Ví dụ: tạo “item8” bắt đầu ở row-line 1 và colum-line 2, và kết thúc ở row-line 5 và column-line 6:

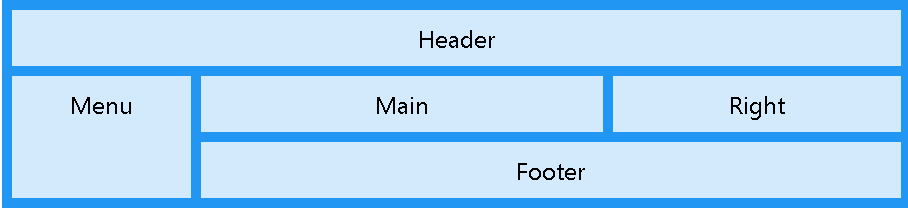
.item8 {  
  grid-area: 1 / 2 / 5 / 6;  
}

Tạo “item 8” bắt đầu ở row-line 2 và column-line 1 và trải dài 2 hàng và 3 cột:

.item8 {  
  grid-area: 2 / 1 / span 2 / span 3;  
}

* Naming Grid Items

Thuộc tính grid-area có thể được sử dụng để gán tên cho các mục grid.



Các mục grid được đặt tên có thể được gọi bằng thuộc tính template-areas của grid container.

Ví dụ: item 1 có tên là “myArea” và trải dài tất cả 5 cột:

.item1 {  
  grid-area: myArea;  
}  
.grid-container {  
  grid-template-areas: 'myArea myArea myArea myArea myArea';  
}

Mỗi hàng được định nghĩa bởi dấu (‘’)

Các cột trong mỗi hàng được định nghĩa bên trong dấu nháy, cách nhau bởi một dấu cách.

Ví dụ:

.item1 {  
  grid-area: myArea;  
}  
.grid-container {  
  grid-template-areas: 'myArea myArea . . .';  
}

Tạo “item1” trải dài 2 cột và 2 hàng:

.grid-container {  
  grid-template-areas: 'myArea myArea . . .' 'myArea myArea . . .';  
}

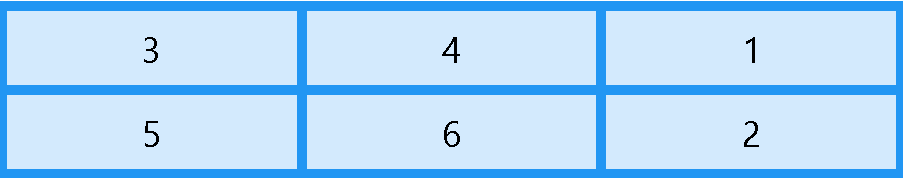
Đặt tên cho tất cả các mục, và tạo một bản mẫu ready-to-use cho trang web.

.item1 { grid-area: header; }  
.item2 { grid-area: menu; }  
.item3 { grid-area: main; }  
.item4 { grid-area: right; }  
.item5 { grid-area: footer; }  
  
.grid-container {  
  grid-template-areas:  
    'header header header header header header'  
    'menu main main main right right'  
    'menu footer footer footer footer footer';  
}

* The Order of the Items

Bố cục grid cho phép chúng ta xác định vị trí các mục ở bất cứ nơi nào chúng ta muốn.

Mục đầu tiên trong mã HTML không nhất thiêt phải xuất hiện như mục đầu tiên trong grid.



Ví dụ:

.item1 { grid-area: 1 / 3 / 2 / 4; }  
.item2 { grid-area: 2 / 3 / 3 / 4; }  
.item3 { grid-area: 1 / 1 / 2 / 2; }  
.item4 { grid-area: 1 / 2 / 2 / 3; }  
.item5 { grid-area: 2 / 1 / 3 / 2; }  
.item6 { grid-area: 2 / 2 / 3 / 3; }

Bạn có thể sắp xếp lại thứ tự cho các kích cỡ màn hình nào đó, bằng cách sử dụng truy vấn phương tiện truyền thông:

@media only screen and (max-width: 500px) {  
  .item1 { grid-area: 1 / span 3 / 2 / 4; }  
  .item2 { grid-area: 3 / 3 / 4 / 4; }  
  .item3 { grid-area: 2 / 1 / 3 / 2; }  
  .item4 { grid-area: 2 / 2 / span 2 / 3; }  
  .item5 { grid-area: 3 / 1 / 4 / 2; }  
  .item6 { grid-area: 2 / 3 / 3 / 4; }  
}

**2.3.5 References**

**2.3.5.1. CSS References**

*1.*Thuộc tính CSS

## 

|  |  |
| --- | --- |
| align-content | Specifies the alignment between the lines inside a flexible container when the items do not use all available space |
| align-items | Specifies the alignment for items inside a flexible container |
| align-self | Specifies the alignment for selected items inside a flexible container |
| all | Resets all properties (except unicode-bidi and direction) |
| animation | A shorthand property for all the animation-\* properties |
| animation-delay | Specifies a delay for the start of an animation |
| animation-direction | Specifies whether an animation should be played forwards, backwards or in alternate cycles |
| animation-duration | Specifies how long an animation should take to complete one cycle |
| animation-fill-mode | Specifies a style for the element when the animation is not playing (before it starts, after it ends, or both) |
| animation-iteration-count | Specifies the number of times an animation should be played |
| animation-name | Specifies a name for the @keyframes animation |
| animation-play-state | Specifies whether the animation is running or paused |
| animation-timing-function | Specifies the speed curve of an animation |

## 

|  |  |
| --- | --- |
| backface-visibility | Defines whether or not the back face of an element should be visible when facing the user |
| background | A shorthand property for all the background-\* properties |
| background-attachment | Sets whether a background image scrolls with the rest of the page, or is fixed |
| background-blend-mode | Specifies the blending mode of each background layer (color/image) |
| background-clip | Defines how far the background (color or image) should extend within an element |
| background-color | Specifies the background color of an element |
| background-image | Specifies one or more background images for an element |
| background-origin | Specifies the origin position of a background image |
| background-position | Specifies the position of a background image |
| background-repeat | Sets if/how a background image will be repeated |
| background-size | Specifies the size of the background images |
| border | A shorthand property for border-width, border-style and border-color |
| border-bottom | A shorthand property for border-bottom-width, border-bottom-style and border-bottom-color |
| border-bottom-color | Sets the color of the bottom border |
| border-bottom-left-radius | Defines the radius of the border of the bottom-left corner |
| border-bottom-right-radius | Defines the radius of the border of the bottom-right corner |
| border-bottom-style | Sets the style of the bottom border |
| border-bottom-width | Sets the width of the bottom border |
| border-collapse | Sets whether table borders should collapse into a single border or be separated |
| border-color | Sets the color of the four borders |
| border-image | A shorthand property for all the border-image-\* properties |
| border-image-outset | Specifies the amount by which the border image area extends beyond the border box |
| border-image-repeat | Specifies whether the border image should be repeated, rounded or stretched |
| border-image-slice | Specifies how to slice the border image |
| border-image-source | Specifies the path to the image to be used as a border |
| border-image-width | Specifies the width of the border image |
| border-left | A shorthand property for all the border-left-\* properties |
| border-left-color | Sets the color of the left border |
| border-left-style | Sets the style of the left border |
| border-left-width | Sets the width of the left border |
| border-radius | A shorthand property for the four border-\*-radius properties |
| border-right | A shorthand property for all the border-right-\* properties |
| border-right-color | Sets the color of the right border |
| border-right-style | Sets the style of the right border |
| border-right-width | Sets the width of the right border |
| border-spacing | Sets the distance between the borders of adjacent cells |
| border-style | Sets the style of the four borders |
| border-top | A shorthand property for border-top-width, border-top-style and border-top-color |
| border-top-color | Sets the color of the top border |
| border-top-left-radius | Defines the radius of the border of the top-left corner |
| border-top-right-radius | Defines the radius of the border of the top-right corner |
| border-top-style | Sets the style of the top border |
| border-top-width | Sets the width of the top border |
| border-width | Sets the width of the four borders |
| bottom | Sets the elements position, from the bottom of its parent element |
| box-decoration-break | Sets the behavior of the background and border of an element at page-break, or, for in-line elements, at line-break. |
| box-shadow | Attaches one or more shadows to an element |
| box-sizing | Defines how the width and height of an element are calculated: should they include padding and borders, or not |
| break-after | Specifies the page-, column-, or region-break behavior after the generated box |
| break-before | Specifies the page-, column-, or region-break behavior before the generated box |
| break-inside | Specifies the page-, column-, or region-break behavior inside the generated box |

## 

|  |  |
| --- | --- |
| caption-side | Specifies the placement of a table caption |
| caret-color | Specifies the color of the cursor (caret) in inputs, textareas, or any element that is editable |
| @charset | Specifies the character encoding used in the style sheet |
| clear | Specifies on which sides of an element floating elements are not allowed to float |
| clip | Clips an absolutely positioned element |
| color | Sets the color of text |
| column-count | Specifies the number of columns an element should be divided into |
| column-fill | Specifies how to fill columns, balanced or not |
| column-gap | Specifies the gap between the columns |
| column-rule | A shorthand property for all the column-rule-\* properties |
| column-rule-color | Specifies the color of the rule between columns |
| column-rule-style | Specifies the style of the rule between columns |
| column-rule-width | Specifies the width of the rule between columns |
| column-span | Specifies how many columns an element should span across |
| column-width | Specifies the column width |
| columns | A shorthand property for column-width and column-count |
| content | Used with the :before and :after pseudo-elements, to insert generated content |
| counter-increment | Increases or decreases the value of one or more CSS counters |
| counter-reset | Creates or resets one or more CSS counters |
| cursor | Specifies the mouse cursor to be displayed when pointing over an element |

## 

|  |  |
| --- | --- |
| direction | Specifies the text direction/writing direction |
| display | Specifies how a certain HTML element should be displayed |

## 

|  |  |
| --- | --- |
| empty-cells | Specifies whether or not to display borders and background on empty cells in a table |

## 

|  |  |
| --- | --- |
| filter | Defines effects (e.g. blurring or color shifting) on an element before the element is displayed |
| flex | A shorthand property for the flex-grow, flex-shrink, and the flex-basis properties |
| flex-basis | Specifies the initial length of a flexible item |
| flex-direction | Specifies the direction of the flexible items |
| flex-flow | A shorthand property for the flex-direction and the flex-wrap properties |
| flex-grow | Specifies how much the item will grow relative to the rest |
| flex-shrink | Specifies how the item will shrink relative to the rest |
| flex-wrap | Specifies whether the flexible items should wrap or not |
| float | Specifies whether or not a box should float |
| font | A shorthand property for the font-style, font-variant, font-weight, font-size/line-height, and the font-family properties |
| @font-face | A rule that allows websites to download and use fonts other than the "web-safe" fonts |
| font-family | Specifies the font family for text |
| font-feature-settings | Allows control over advanced typographic features in OpenType fonts |
| @font-feature-values | Allows authors to use a common name in font-variant-alternate for feature activated differently in OpenType |
| font-kerning | Controls the usage of the kerning information (how letters are spaced) |
| font-language-override | Controls the usage of language-specific glyphs in a typeface |
| font-size | Specifies the font size of text |
| font-size-adjust | Preserves the readability of text when font fallback occurs |
| font-stretch | Selects a normal, condensed, or expanded face from a font family |
| font-style | Specifies the font style for text |
| font-synthesis | Controls which missing typefaces (bold or italic) may be synthesized by the browser |
| font-variant | Specifies whether or not a text should be displayed in a small-caps font |
| font-variant-alternates | Controls the usage of alternate glyphs associated to alternative names defined in @font-feature-values |
| font-variant-caps | Controls the usage of alternate glyphs for capital letters |
| font-variant-east-asian | Controls the usage of alternate glyphs for East Asian scripts (e.g Japanese and Chinese) |
| font-variant-ligatures | Controls which ligatures and contextual forms are used in textual content of the elements it applies to |
| font-variant-numeric | Controls the usage of alternate glyphs for numbers, fractions, and ordinal markers |
| font-variant-position | Controls the usage of alternate glyphs of smaller size positioned as superscript or subscript regarding the baseline of the font |
| font-weight | Specifies the weight of a font |

## 

|  |  |
| --- | --- |
| grid | A shorthand property for the grid-template-rows, grid-template-columns, grid-template-areas, grid-auto-rows, grid-auto-columns, and the grid-auto-flow properties |
| grid-area | Either specifies a name for the grid item, or this property is a shorthand property for the grid-row-start, grid-column-start, grid-row-end, and grid-column-end properties |
| grid-auto-columns | Specifies a default column size |
| grid-auto-flow | Specifies how auto-placed items are inserted in the grid |
| grid-auto-rows | Specifies a default row size |
| grid-column | A shorthand property for the grid-column-start and the grid-column-end properties |
| grid-column-end | Specifies where to end the grid item |
| grid-column-gap | Specifies the size of the gap between columns |
| grid-column-start | Specifies where to start the grid item |
| grid-gap | A shorthand property for the grid-row-gap and grid-column-gap properties |
| grid-row | A shorthand property for the grid-row-start and the grid-row-end properties |
| grid-row-end | Specifies where to end the grid item |
| grid-row-gap | Specifies the size of the gap between rows |
| grid-row-start | Specifies where to start the grid item |
| grid-template | A shorthand property for the grid-template-rows, grid-template-columns and grid-areas properties |
| grid-template-areas | Specifies how to display columns and rows, using named grid items |
| grid-template-columns | Specifies the size of the columns, and how many columns in a grid layout |
| grid-template-rows | Specifies the size of the rows in a grid layout |

## 

|  |  |
| --- | --- |
| hanging-punctuation | Specifies whether a punctuation character may be placed outside the line box |
| height | Sets the height of an element |
| hyphens | Sets how to split words to improve the layout of paragraphs |

## 

|  |  |
| --- | --- |
| image-rendering | Gives a hint to the browser about what aspects of an image are most important to preserve when the image is scaled |
| @import | Allows you to import a style sheet into another style sheet |
| isolation | Defines whether an element must create a new stacking content |

## 

|  |  |
| --- | --- |
| justify-content | Specifies the alignment between the items inside a flexible container when the items do not use all available space |

## 

|  |  |
| --- | --- |
| @keyframes | Specifies the animation code |

## 

|  |  |
| --- | --- |
| left | Specifies the left position of a positioned element |
| letter-spacing | Increases or decreases the space between characters in a text |
| line-break | Specifies how/if to break lines |
| line-height | Sets the line height |
| list-style | Sets all the properties for a list in one declaration |
| list-style-image | Specifies an image as the list-item marker |
| list-style-position | Specifies the position of the list-item markers (bullet points) |
| list-style-type | Specifies the type of list-item marker |

## 

|  |  |
| --- | --- |
| margin | Sets all the margin properties in one declaration |
| margin-bottom | Sets the bottom margin of an element |
| margin-left | Sets the left margin of an element |
| margin-right | Sets the right margin of an element |
| margin-top | Sets the top margin of an element |
| max-height | Sets the maximum height of an element |
| max-width | Sets the maximum width of an element |
| @media | Sets the style rules for different media types/devices/sizes |
| min-height | Sets the minimum height of an element |
| min-width | Sets the minimum width of an element |
| mix-blend-mode | Specifies how an element's content should blend with its direct parent background |

## 

|  |  |
| --- | --- |
| object-fit | Specifies how the contents of a replaced element should be fitted to the box established by its used height and width |
| object-position | Specifies the alignment of the replaced element inside its box |
| opacity | Sets the opacity level for an element |
| order | Sets the order of the flexible item, relative to the rest |
| orphans | Sets the minimum number of lines that must be left at the bottom of a page when a page break occurs inside an element |
| outline | A shorthand property for the outline-width, outline-style, and the outline-color properties |
| outline-color | Sets the color of an outline |
| outline-offset | Offsets an outline, and draws it beyond the border edge |
| outline-style | Sets the style of an outline |
| outline-width | Sets the width of an outline |
| overflow | Specifies what happens if content overflows an element's box |
| overflow-wrap | Specifies whether or not the browser may break lines within words in order to prevent overflow (when a string is too long to fit its containing box) |
| overflow-x | Specifies whether or not to clip the left/right edges of the content, if it overflows the element's content area |
| overflow-y | Specifies whether or not to clip the top/bottom edges of the content, if it overflows the element's content area |

## 

|  |  |
| --- | --- |
| padding | A shorthand property for all the padding-\* properties |
| padding-bottom | Sets the bottom padding of an element |
| padding-left | Sets the left padding of an element |
| padding-right | Sets the right padding of an element |
| padding-top | Sets the top padding of an element |
| page-break-after | Sets the page-break behavior after an element |
| page-break-before | Sets the page-break behavior before an element |
| page-break-inside | Sets the page-break behavior inside an element |
| perspective | Gives a 3D-positioned element some perspective |
| perspective-origin | Defines at which position the user is looking at the 3D-positioned element |
| pointer-events | Defines whether or not an element reacts to pointer events |
| position | Specifies the type of positioning method used for an element (static, relative, absolute or fixed) |

## 

|  |  |
| --- | --- |
| quotes | Sets the type of quotation marks for embedded quotations |

## 

|  |  |
| --- | --- |
| resize | Defines if (and how) an element is resizable by the user |
| right | Specifies the right position of a positioned element |

## 

|  |  |
| --- | --- |
| scroll-behavior | Specifies whether to smoothly animate the scroll position in a scrollable box, instead of a straight jump |

## 

|  |  |
| --- | --- |
| tab-size | Specifies the width of a tab character |
| table-layout | Defines the algorithm used to lay out table cells, rows, and columns |
| [text-align](https://www.w3schools.com/cssref/pr_text_text-align.asp) | Specifies the horizontal alignment of text |
| text-align-last | Describes how the last line of a block or a line right before a forced line break is aligned when text-align is "justify" |
| text-combine-upright | Specifies the combination of multiple characters into the space of a single character |
| text-decoration | Specifies the decoration added to text |
| text-decoration-color | Specifies the color of the text-decoration |
| text-decoration-line | Specifies the type of line in a text-decoration |
| text-decoration-style | Specifies the style of the line in a text decoration |
| text-indent | Specifies the indentation of the first line in a text-block |
| text-justify | Specifies the justification method used when text-align is "justify" |
| text-orientation | Defines the orientation of the text in a line |
| [text-overflow](https://www.w3schools.com/cssref/css3_pr_text-overflow.asp) | Specifies what should happen when text overflows the containing element |
| [text-shadow](https://www.w3schools.com/cssref/css3_pr_text-shadow.asp) | Adds shadow to text |
| [text-transform](https://www.w3schools.com/cssref/pr_text_text-transform.asp) | Controls the capitalization of text |
| text-underline-position | Specifies the position of the underline which is set using the text-decoration property |
| [top](https://www.w3schools.com/cssref/pr_pos_top.asp) | Specifies the top position of a positioned element |
| [transform](https://www.w3schools.com/cssref/css3_pr_transform.asp) | Applies a 2D or 3D transformation to an element |
| [transform-origin](https://www.w3schools.com/cssref/css3_pr_transform-origin.asp) | Allows you to change the position on transformed elements |
| [transform-style](https://www.w3schools.com/cssref/css3_pr_transform-style.asp) | Specifies how nested elements are rendered in 3D space |
| [transition](https://www.w3schools.com/cssref/css3_pr_transition.asp) | A shorthand property for all the transition-\* properties |
| [transition-delay](https://www.w3schools.com/cssref/css3_pr_transition-delay.asp) | Specifies when the transition effect will start |
| [transition-duration](https://www.w3schools.com/cssref/css3_pr_transition-duration.asp) | Specifies how many seconds or milliseconds a transition effect takes to complete |
| [transition-property](https://www.w3schools.com/cssref/css3_pr_transition-property.asp) | Specifies the name of the CSS property the transition effect is for |
| [transition-timing-function](https://www.w3schools.com/cssref/css3_pr_transition-timing-function.asp) | Specifies the speed curve of the transition effect |

## 

|  |  |
| --- | --- |
| [unicode-bidi](https://www.w3schools.com/cssref/pr_text_unicode-bidi.asp) | Used together with the [direction](https://www.w3schools.com/cssref/pr_text_direction.asp) property to set or return whether the text should be overridden to support multiple languages in the same document |
| [user-select](https://www.w3schools.com/cssref/css3_pr_user-select.asp) | Specifies whether the text of an element can be selected |

## 

|  |  |
| --- | --- |
| [vertical-align](https://www.w3schools.com/cssref/pr_pos_vertical-align.asp) | Sets the vertical alignment of an element |
| [visibility](https://www.w3schools.com/cssref/pr_class_visibility.asp) | Specifies whether or not an element is visible |

## 

|  |  |
| --- | --- |
| [white-space](https://www.w3schools.com/cssref/pr_text_white-space.asp) | Specifies how white-space inside an element is handled |
| widows | Sets the minimum number of lines that must be left at the top of a page when a page break occurs inside an element |
| [width](https://www.w3schools.com/cssref/pr_dim_width.asp) | Sets the width of an element |
| [word-break](https://www.w3schools.com/cssref/css3_pr_word-break.asp) | Specifies how words should break when reaching the end of a line |
| [word-spacing](https://www.w3schools.com/cssref/pr_text_word-spacing.asp) | Increases or decreases the space between words in a text |
| [word-wrap](https://www.w3schools.com/cssref/css3_pr_word-wrap.asp) | Allows long, unbreakable words to be broken and wrap to the next line |
| [writing-mode](https://www.w3schools.com/cssref/css3_pr_writing-mode.asp) | Specifies whether lines of text are laid out horizontally or vertically |

## 

|  |  |
| --- | --- |
| [z-index](https://www.w3schools.com/cssref/pr_pos_z-index.asp) | Sets the stack order of a positioned element |

**2.3.5.4. Chức năng CSS**

Các hàm CSS được sử dụng làm giá trị cho các thuộc tính CSS khác nhau.

|  |  |
| --- | --- |
| **Function** | **Description** |
| [attr()](https://www.w3schools.com/cssref/func_attr.asp) | Returns the value of an attribute of the selected element |
| [calc()](https://www.w3schools.com/cssref/func_calc.asp) | Allows you to perform calculations to determine CSS property values |
| [cubic-bezier()](https://www.w3schools.com/cssref/func_cubic-bezier.asp) | Defines a Cubic Bezier curve |
| [hsl()](https://www.w3schools.com/cssref/func_hsl.asp) | Defines colors using the Hue-Saturation-Lightness model (HSL) |
| [hsla()](https://www.w3schools.com/cssref/func_hsla.asp) | Defines colors using the Hue-Saturation-Lightness-Alpha model (HSLA) |
| [linear-gradient()](https://www.w3schools.com/cssref/func_linear-gradient.asp) | Sets a linear gradient as the background image. Define at least two colors (top to bottom) |
| [radial-gradient()](https://www.w3schools.com/cssref/func_radial-gradient.asp) | Sets a radial gradient as the background image. Define at least two colors (center to edges) |
| [repeating-linear-gradient()](https://www.w3schools.com/cssref/func_repeating-linear-gradient.asp) | Repeats a linear gradient |
| [repeating-radial-gradient()](https://www.w3schools.com/cssref/func_repeating-radial-gradient.asp) | Repeats a radial gradient |
| [rgb()](https://www.w3schools.com/cssref/func_rgb.asp) | Defines colors using the Red-Green-Blue model (RGB) |
| [rgba()](https://www.w3schools.com/cssref/func_rgba.asp) | Defines colors using the Red-Green-Blue-Alpha model (RGBA) |
| [var()](https://www.w3schools.com/cssref/func_var.asp) | Inserts the value of a custom property |

***2.3.5.5 CSS web safe fonts***

**Kết hợp phông chữ thường được sử dụng**

Thuộc tính họ phông chữ phải giữ một số tên phông chữ dưới dạng hệ thống "dự phòng", để đảm bảo khả năng tương thích tối đa giữa các trình duyệt / hệ điều hành. Nếu trình duyệt không hỗ trợ phông chữ đầu tiên, nó sẽ thử phông chữ tiếp theo.

Bắt đầu với phông chữ bạn muốn và kết thúc bằng một họ chung, để cho phép trình duyệt chọn một phông chữ tương tự trong họ chung, nếu không có phông chữ nào khác:

Vd

1. <!DOCTYPE html**>**
2. **<html>**
3. **<head>**
4. **<style>**
5. p.a {
6. font-family: "Times New Roman", Times, serif;
7. }
8. p.b {
9. font-family: Arial, Helvetica, sans-serif;
10. }
11. **</style>**
12. **</head>**
13. **<body>**
14. **<h1>**The font-family Property**</h1>**
15. **<p** class="a"**>**This is a paragraph, shown in the Times New Roman font.**</p>**
16. **<p** class="b"**>**This is a paragraph, shown in the Arial font.**</p>**
17. **</body>**
18. **</html**

**2.3.5.6 CSS Animatable**

Một số thuộc tính CSS là *animatable* , có nghĩa là chúng có thể được sử dụng trong hình ảnh động và hiệu ứng chuyển tiếp.

Thuộc tính hoạt hình có thể thay đổi dần dần từ giá trị này sang giá trị khác, như kích thước, số, tỷ lệ phần trăm và màu sắc.

Vi du :

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <style>
5. #myDIV {
6. **width**: 300px;
7. **height**: 200px;
8. **background**: red;
9. -webkit-animation: mymove 5s infinite; /\* Chrome, Safari, Opera \*/
10. animation: mymove 5s infinite;
11. }/\* Chrome, Safari, Opera \*/
12. @-webkit-keyframes mymove {
13. from {**background-color**: red;}
14. to {**background-color**: blue;}
15. }/\* Standard syntax \*/
16. @keyframes mymove {
17. from {**background-color**: red;}
18. to {**background-color**: blue;}
19. }
20. </style>
21. </head>
22. <body>
23. <h1>Animation of background-color</h1>
24. <p>Gradually change the background-color from red to blue:<p>
25. <div id="myDIV"></div>
26. <p><strong>Note:</strong> CSS Animations do not work in Internet Explorer 9 and earlier versions.</p>
27. </body>
28. </html>

Cấu trúc chung: animation: giá trị;

* Trên firefox: -moz-animation: giá trị;
* Trên Google Chrome và Safari: -webkit-animation: giá trị;
* Trên: Opera: -o-animation: giá trị;

|  |  |
| --- | --- |
| **Thuộc tính** | **Giải thích** |
| animation-name | Xác định tên cho một animation. |
| animation-duration | Xác định thời gian để hoàn thành một chuyển động, mặc định là 0s. |
| animation-timing-function | Thời gian chuyển động |
| animation-delay | Thời gian delay của chuyển động |
| animation-iteration-count | Số lần thực hiện chuyển động |
| animation-direction | Hướng chuyển động |
| animation-fill-mode | Xác định chế độ điền (di chuyển đến và dừng tại điểm yêu cầu của giá trị) |
| animation-play-state | Xác định dừng – chạy |

animation-timing-function là thuộc tính mô tả cách thức thực hiện theo thời gian. Một số giá trị của animation-timing-function phổ biến như sau:

* linear: Chuyển động đều tốc độ trong suốt chu kỳ.
* ease: Chuyển động lúc đầu chậm, sau đó nhanh dần, và sau đó từ từ kết thúc.
* ease-in: Chuyển động ban đầu sẽ chậm, và sau đó nhanh dần lên..
* ease-out: Chuyển động ngược lại bên trên.
* ease-in-out: Tương tự chuyển động của ease
* cubic-bezier(n,n,n,n): Giá trị riêng cho chuyển động, (n = 0 hoặc 1), với 1 nhanh, 0 chậm.

**Thuộc tính animation-iteration-count**

Là thuộc tính thể hiện số lần chuyển động được thực hiện, Có 2 giá trị cho thuộc tính này đó là số nguyên và infinite (không giới hạn).

ví dụ animation-iteration-count: 4; hoặc animation-iteration-count: infinite;

**Thuộc tính animation-direction**

Thuộc tính này chỉ định hướng đi cho đối tượng, với các giá trị như sau:

* normal: Hoạt động bình thường theo mặc định.
* reverse: Hoạt động với chiều ngược lại mặc định.
* alternate: Đảo chiều sau một chu kỳ hoạt động.
* alternate-reverse: tương tự alternate nhưng trái 1 nhịp.

**Thuộc tính animation-fill-mode**

Thuộc tính này sẽ xác định vị trí cuối cùng của hoạt động. ví dụ như di chuyển đến và đứng tại đó, di chuyển đến cuối chu kỳ và quay trở lại vị trí cũ và đứng tại đó. Các giá trị của thuộc tính này như sau

* forwards: Di chuyển đến và ở tại vị trí cuối cùng trong chu kỳ chuyển động.
* backwards: Trở lại vị trí đầu tiên sau khi kết thúc chu kỳ.
* both: Mình không rõ thuộc tính này, dùng thì không khác gì forwards cả nhưng theo W3C thì nó sẽ kết hợp cả 2.

**Thuộc tính animation-play-state**

Thuộc tính này sẽ điều khiển việc bắt đầu hoạt dừng lại hoạt động. Thuộc tính có 2 giá trị là paused và running

ví dụ: animation-play-state: running;

ong 2 giây. Nó sẽ thực hiện chuyển động dựa trên keyframes out từ trái sang phải với khoảng cách là 200px.

**2.3.5.7 CSS Units**

**Unit trong CSS là gì?**

Unit trong CSS là đơn vị đo của các thuộc tính trong CSS như margin, padding, width, height...

Giá trị có thể mang giá trị âm, không có khoảng trắng giữa giá trị và đơn vị đo. Ví dụ: 20px, 30em ...

Trong CSS có 2 loại đơn vị đó là đơn vị tuyệt đối (absolute) và đơn vị tương đối (relative). Mình sẽ cùng tìm hiểu ý nghĩa của từng loại đơn vị trong mục tiếp theo.

Ý nghĩa của từng loại đơn vị (unit)

**Absolute**

Absolute là loại đơn vị có giá trị không thay đổi dù cho bất cứ kỳ đâu, nhưng chỉ khi thiết bị đầu ra của bạn có độ phân giải đủ cao. Trên máy in laser, 1cm phải chính xác 1 centimet (centimet của thước đo vật lý).

Nhưng trên các thiết bị có độ phân giải thấp, như màn hình máy tính, màn hình điện thoại ... CSS không yêu cầu độ chính xác 100%.

Và vì vậy, kết quả có xu hướng khác nhau giữa các thiết bị. Tốt hơn là nên dành các đơn vị này cho các thiết bị có độ phân giải cao và đặc biệt cho đầu ra là máy in. Trên màn hình máy tính và thiết bị cầm tay, có thể bạn sẽ không nhận được những gì bạn mong đợi.

Một số đơn vị tuyệt đối (absolute) trong CSS như sau:

• cm: centimet

• mm: milimet

• in: inch, 1in = 96px = 2.54 cm

• px: 1px tương ứng với một điểm ảnh trên máy tính. 1px = 1/96 in

• pt: 1pt =1/71in

• pc: 1pc =12pt

ví dụ

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <style>
5. div.cm {**font-size**: 1cm}
6. div.mm {**font-size**: 10mm}
7. div.in {**font-size**: 0.5in}
8. div.px {**font-size**: 10px}
9. div.pt {**font-size**: 10pt}
10. div.pc {**font-size**: 1pc}
11. </style>
12. </head>
13. <body>
14. <div class="cm">Đây là đơn vị 1cm</div>
15. <div class="mm">Đây là đơn vị 10mm</div>
16. <div class="in">Đây là đơn vị 0.5in</div>
17. <div class="px">Đây là đơn vị 10px</div>
18. <div class="pt">Đây là đơn vị 1pt</div>
19. <div class="pc">Đây là đơn vị 1pc</div>
20. </body>
21. </html>

Từ kết quả trên cho ta cái nhìn tổng quát về sự khác nhau giữa các đơn vị đo như cm, mm, pt, pc ... và đơn vị đo này nhỏ hơn so với đơn vị trong thực tế. Đó là do màn hình máy tính của mình có độ phân giải không đủ cao nên không chính xác 100% như trong thước đo thực tế được.

**Relative**

Relative là loại đơn vị sẽ có giá trị tương đối so với độ dài của thuộc tính.

Một số đơn vị tương đối (relative) trong CSS như sau:

• em: Tương đối so với font-size của phần tử cha.

• ex: Tương đối so với x-height của font hiện tại.

• ch: Relative to width of the "0" (zero)

• rem: Tương đối so với phần tử gốc. Phần tử gốc ở đây là thẻ html.

• vw: Tương đối 1% width của kích thước cửa sổ trình duyệt (viewport).

• vh: Tương đối 1% height của kích thước cửa sổ trình duyệt (viewport).

• vmin Relative to 1% của kích thước cửa sổ trình duyệt nhỏ hơn

• vmax Relative to 1% của kích thước cửa sổ trình duyệt lớn hơn

• %: Có giá trị tương đối so với phần tử cha.

Ví dụ:

1. div.parent {
2. **background-color**: red;
3. **width**: 500px;
4. }
5. div.child {
6. **border**: 1px solid black;
7. **background-color**: yellow;
8. }

***2.3.5.8 CSS PX-EM Converter***

Như đã được trình bày trong chương **Đơn vị trong CSS**, một đơn vị được sử dụng cho chiều cao của font. Một đơn vị em tương đương với kích cỡ của một font đã cho, nếu bạn gán kích cỡ của font là 12px thì mỗi đơn vị em sẽ là 12px, do đó 2em sẽ là 24px.

Do đó, trong phần chuyển đổi dưới đây, khi bạn thiết lập kích cỡ bằng Pixel mặc định là giá trị nào, thì đó chính là kích cỡ của font bạn đang sử dụng, và cũng chính là giá trị tương đương 1 em.

Chuyển đổi PX sang EM

* Đặt kích thước phông chữ mặc định (thường là 16px)
* Sau đó, chuyển đổi giá trị pixel thành em, dựa trên kích thước mặc định
* Hoặc, chuyển đổi giá trị em thành pixel, dựa trên kích thước mặc định

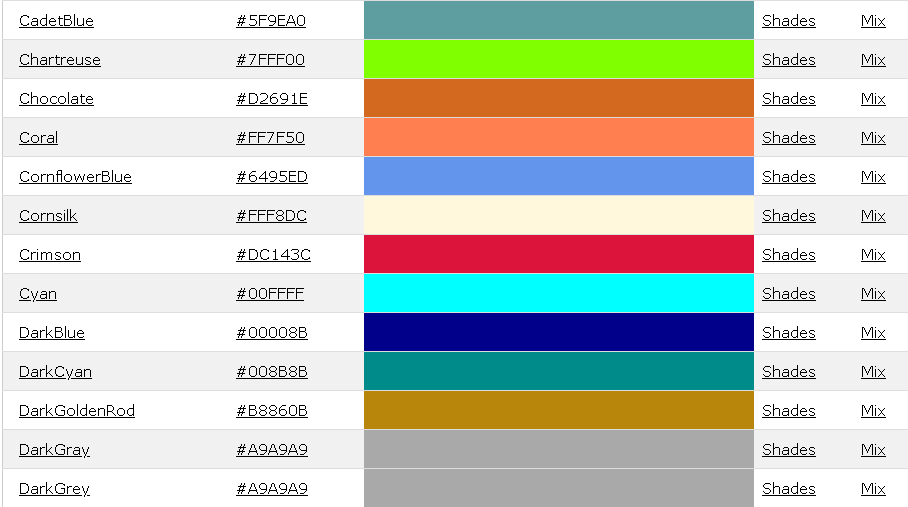
***2.3.5.9 CSS Colors***

**CSS colors**

* Tên màu sắc được hỗ trợ bởi tất cả các trình duyệt

Tất cả các trình duyệt hiện đại hỗ trợ 140 tên màu:

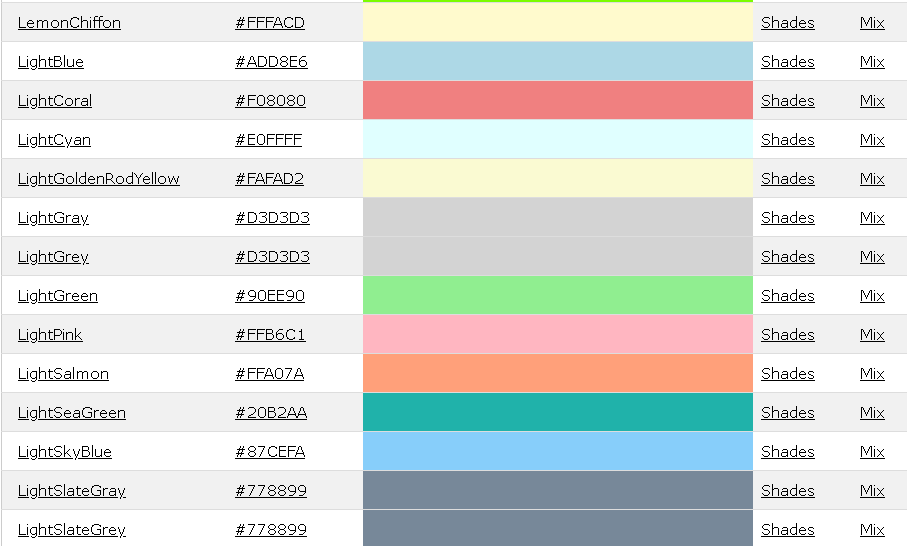




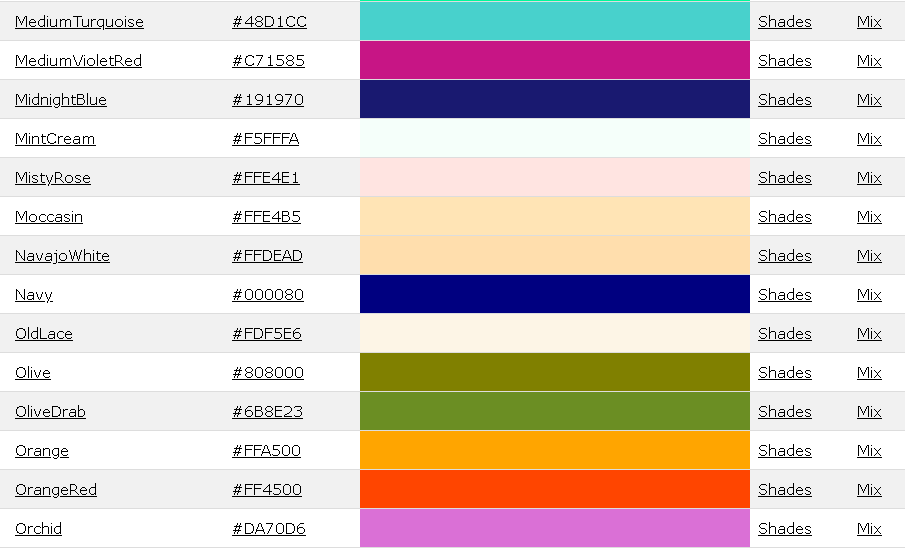




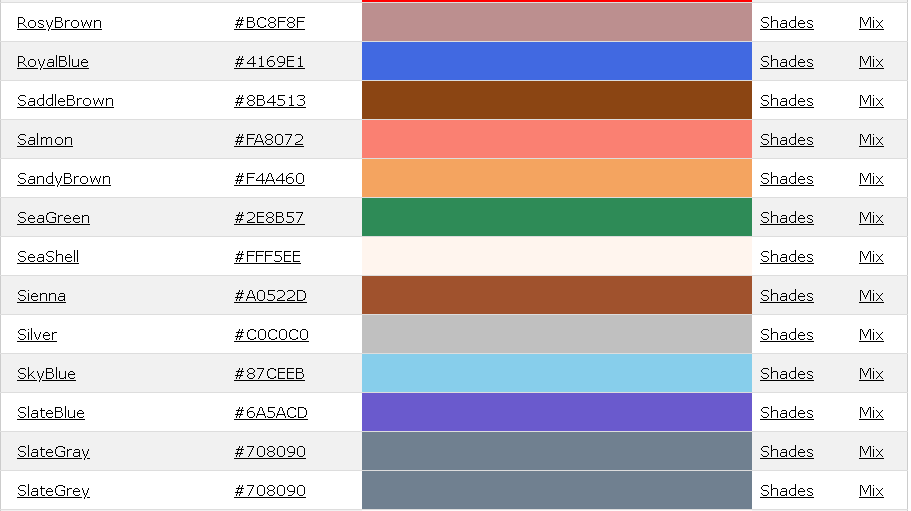














***2.3.5.10 CSS Colors Values***

* **CSS colors**

Màu trong CSS có thể được xác định bằng các phương thức:

* Hexadecimal colors
* RGB colors
* RGBA colors
* HSL colors
* HSLA colors
* Predefined/Cross-browser color names
* Hexadecimal Colors

Một màu hexadecimal được quy định với #RRGGBB, trong đó các số nguyên hexadecimal RR(màu đỏ), GG(màu xanh) và BB(màu xanh dương) chỉ định các thành phần của màu. Tất cả các giá trị phải nằm trong khoảng từ 00 đến FF.

Ví dụ, giá trị #0000ff được hiển thị là màu xanh lam, vì thành phần màu xanh lam được đặt thành giá trị cao nhất(ff) và các giá trị khác được đặt thành 00.

#p1 {background-color: #ff0000;}   /\* red \*/  
#p2 {background-color: #00ff00;}   /\* green \*/  
#p3 {background-color: #0000ff;}   /\* blue \*/

* RGBcolors  
  Giá trị màu RGB được chỉ định với hàm rgb (), có cú pháp sau: rgb (đỏ, xanh lá cây, xanh dương) .

Mỗi tham số (đỏ, lục và lam) xác định cường độ của màu và có thể là một số nguyên trong khoảng từ 0 đến 255 hoặc giá trị phần trăm (từ 0% đến 100%).

Ví dụ: giá trị rgb (0,0,255) được hiển thị là màu xanh lam, bởi vì tham số màu xanh được đặt thành giá trị cao nhất của nó (255) và các giá trị khác được đặt thành 0.

Ngoài ra, các giá trị sau xác định màu bằng nhau: rgb (0,0,255) và rgb (0%, 0%, 100%).

Ví dụ:

#p1 {background-color: rgb(255, 0, 0);}   /\* red \*/  
#p2 {background-color: rgb(0, 255, 0);}   /\* green \*/  
#p3 {background-color: rgb(0, 0, 255);}   /\* blue \*/

* RGBAColors  
  Giá trị màu RGBA là phần mở rộng của giá trị màu RGB với kênh alpha - chỉ định độ mờ của đối tượng.

Màu RGBA được chỉ định với hàm rgba (), có cú pháp sau: rgba (đỏ, xanh lá cây, xanh dương, alpha)

Tham số alpha là một số trong khoảng 0,0 (hoàn toàn trong suốt) và 1,0 (hoàn toàn mờ).

#p1 {background-color: rgba(255, 0, 0, 0.3);}   /\* red with opacity \*/  
#p2 {background-color: rgba(0, 255, 0, 0.3);}   /\* green with opacity \*/  
#p3 {background-color: rgba(0, 0, 255, 0.3);}   /\* blue with opacity \*/

* HSL Colors

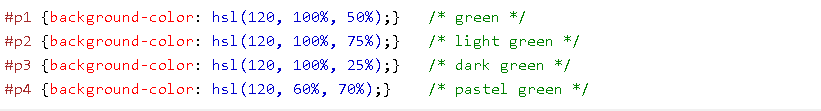
HSL là viết tắt của màu sắc, độ bão hòa và độ sáng - và đại diện cho màu đại diện phối hợp hình trụ của màu sắc.

Giá trị màu HSL được chỉ định với hàm hsl (), có cú pháp sau: *hsl (màu sắc, độ bão hòa, độ sáng)*

Hue là một độ trên bánh xe màu (từ 0 đến 360) - 0 (hoặc 360) là màu đỏ, 120 là màu xanh lá cây, 240 là màu xanh.

Saturation là một giá trị phần trăm; 0% có nghĩa là một màu xám và 100% là màu đầy đủ.

Độ nhẹ cũng là một tỷ lệ phần trăm; 0% là màu đen, 100% là màu trắng.



* HSLA colors

Các giá trị màu HSLA là phần mở rộng của các giá trị màu HSL với kênh alpha - chỉ định độ mờ của đối tượng.

Giá trị màu HSLA được chỉ định với hàm hsla (), có cú pháp sau: *hsla (màu sắc, độ bão hòa, độ sáng, alpha)*

Tham số alpha là một số trong khoảng 0,0 (hoàn toàn trong suốt) và 1,0 (hoàn toàn mờ).

#p1 {background-color: hsla(120, 100%, 50%, 0.3);}   /\* green with opacity \*/  
#p2 {background-color: hsla(120, 100%, 75%, 0.3);}   /\* light green with opacity \*/  
#p3 {background-color: hsla(120, 100%, 25%, 0.3);}   /\* dark green with opacity \*/  
#p4 {background-color: hsla(120, 60%, 70%, 0.3);}    /\* pastel green with opacity \*/

***2.3.5.11 CSS Default Values***

* Default CSS values for HTML elements

|  |  |
| --- | --- |
| **Phần tử** | **Giá trị CSS mặc định** |
| a:link | color: (internal value); text-decoration: underline; cursor: auto; |
| a:visited | color: (internal value); text-decoration: underline; cursor: auto; |
| a:link:active | color: (internal value); |
| a:visited:active | color: (internal value); |
| abbr | None. |
| address | display: block; font-style: italic; |
| area | display: none; |
| article | display: block; |
| aside | display: block; |
| audio | None. |
| b | font-weight: bold; |
| base | None. |
| bdi | None. |
| bdo | unicode-bidi: bidi-override; |
| blockquote | display: block; margin-top: 1em; margin-bottom: 1em; margin-left: 40px; margin-right: 40px; |
| body | display: block; margin: 8px; |
| body:focus | outline: none; |
| br | None. |
| button | None |
| canvas | None. |
| caption | display: table-caption; text-align: center; |
| cite | font-style: italic; |
| code | font-family: monospace; |
| col | display: table-column; |
| colgroup | display: table-column-group |
| datalist | display: none; |
| dd | display: block; margin-left: 40px; |
| del | text-decoration: line-through; |
| details | display: block; |
| dfn | font-style: italic; |
| dialog | None. |
| div | display: block; |
| dl | display: block; margin-top: 1em; margin-bottom: 1em; margin-left: 0; margin-right: 0; |
| dt | display: block; |
| em | font-style: italic; |
| embed:focus | outline: none; |
| fieldset | display: block; margin-left: 2px; margin-right: 2px; padding-top: 0.35em; padding-bottom: 0.625em; padding-left: 0.75em; padding-right: 0.75em; border: 2px groove (internal value); |
| figcaption | display: block; |
| figure | display: block; margin-top: 1em; margin-bottom: 1em; margin-left: 40px; margin-right: 40px; |
| footer | display: block; |
| form | display: block; margin-top: 0em; |
| h1 | display: block; font-size: 2em; margin-top: 0.67em; margin-bottom: 0.67em; margin-left: 0; margin-right: 0; font-weight: bold; |
| h2 | display: block; font-size: 1.5em; margin-top: 0.83em; margin-bottom: 0.83em; margin-left: 0; margin-right: 0; font-weight: bold; |
| h3 | display: block; font-size: 1.17em; margin-top: 1em; margin-bottom: 1em; margin-left: 0; margin-right: 0; font-weight: bold; |
| h4 | display: block; margin-top: 1.33em; margin-bottom: 1.33em; margin-left: 0; margin-right: 0; font-weight: bold; |
| h5 | display: block; font-size: .83em; margin-top: 1.67em; margin-bottom: 1.67em; margin-left: 0; margin-right: 0; font-weight: bold; |
| h6 | display: block; font-size: .67em; margin-top: 2.33em; margin-bottom: 2.33em; margin-left: 0; margin-right: 0; font-weight: bold; |
| head | display: none; |
| header | display: block; |
| hr | display: block; margin-top: 0.5em; margin-bottom: 0.5em; margin-left: auto; margin-right: auto; border-style: inset; border-width: 1px; |
| html | display: block; |
| html:focus | outline: none; |
| i | font-style: italic; |
| iframe:focus | outline: none; |
| iframe[seamless] | display: block; |
| img | display: inline-block; |
| input | None. |
| ins | text-decoration: underline; |
| kbd | font-family: monospace; |
| label | cursor: default; |
| legend | display: block; padding-left: 2px; padding-right: 2px; border: none; |
| li | display: list-item; |
| link | display: none; |
| main | None. |
| map | display: inline; |
| mark | background-color: yellow; color: black; |
| menu | display: block; list-style-type: disc; margin-top: 1em; margin-bottom: 1em; margin-left: 0; margin-right: 0; padding-left: 40px; |
| menuitem | None. |
| meta | None. |
| meter | None. |
| nav | display: block; |
| noscript | None. |
| object:focus | outline: none; |
| ol | display: block; list-style-type: decimal; margin-top: 1em; margin-bottom: 1em; margin-left: 0; margin-right: 0; padding-left: 40px; |
| optgroup | None. |
| option | None. |
| output | display: inline; |
| p | display: block; margin-top: 1em; margin-bottom: 1em; margin-left: 0; margin-right: 0; |
| param | display: none; |
| picture | None. |
| pre | display: block; font-family: monospace; white-space: pre; margin: 1em 0; |
| progress | None. |
| q | display: inline; |
| q::before | content: open-quote; |
| q::after | content: close-quote; |
| rp | None. |
| rt | line-height: normal; |
| ruby | None. |
| s | text-decoration: line-through; |
| samp | font-family: monospace; |
| script | display: none; |
| section | display: block; |
| select | None. |
| small | font-size: smaller; |
| source | None. |
| span | None. |
| strike | text-decoration: line-through; |
| strong | font-weight: bold; |
| style | display: none; |
| sub | vertical-align: sub; font-size: smaller; |
| summary | display: block; |
| sup | vertical-align: super; font-size: smaller; |
| table | display: table; border-collapse: separate; border-spacing: 2px; border-color: gray; |
| tbody | display: table-row-group; vertical-align: middle; border-color: inherit; |
| td | display: table-cell; vertical-align: inherit; |
| template | None. |
| textarea | None. |
| tfoot | display: table-footer-group; vertical-align: middle; border-color: inherit; |
| th | display: table-cell; vertical-align: inherit; font-weight: bold; text-align: center; |
| thead | display: table-header-group; vertical-align: middle; border-color: inherit; |
| time | None. |
| title | display: none; |
| tr | display: table-row; vertical-align: inherit; border-color: inherit; |
| track | None. |
| u | text-decoration: underline; |
| ul | display: block; list-style-type: disc; margin-top: 1em; margin-bottom: 1 em; margin-left: 0; margin-right: 0; padding-left: 40px; |
| var | font-style: italic; |
| video | None. |
| wbr | None. |

***2.3.5.12 CSS Entities***

* Complete CSS Entity Reference

Nếu bạn sử dụng CSS để hiển thị bất kỳ thuộc tính nào trong HTML, bạn có thể sử dụng thực thể CSS được tìm thấy trong bảng dưới đây.

<style>  
h2:after {  
  content: ' \00A7';  
}  
</style>

Tất cả phần tử <h2> sẽ được hiển thị với thuộc tính này ở cuối.

|  |  |  |
| --- | --- | --- |
| **Char** | **CSS Entity** | **Name** |
|  | 0020 | SPACE |
| ! | 0021 | EXCLAMATION MARK |
| " | 0022 | QUOTATION MARK |
| # | 0023 | NUMBER SIGN |
| $ | 0024 | DOLLAR SIGN |
| % | 0025 | PERCENT SIGN |
| & | 0026 | AMPERSAND |
| ' | 0027 | APOSTROPHE |
| ( | 0028 | LEFT PARENTHESIS |
| ) | 0029 | RIGHT PARENTHESIS |
| \* | 002A | ASTERISK |
| + | 002B | PLUS SIGN |
| , | 002C | COMMA |
| - | 002D | HYPHEN-MINUS |
| . | 002E | FULL STOP |
| / | 002F | SOLIDUS |
| 0 | 0030 | DIGIT ZERO |
| 1 | 0031 | DIGIT ONE |
| 2 | 0032 | DIGIT TWO |
| 3 | 0033 | DIGIT THREE |
| 4 | 0034 | DIGIT FOUR |
| 5 | 0035 | DIGIT FIVE |
| 6 | 0036 | DIGIT SIX |
| 7 | 0037 | DIGIT SEVEN |
| 8 | 0038 | DIGIT EIGHT |
| 9 | 0039 | DIGIT NINE |
| : | 003A | COLON |
| ; | 003B | SEMICOLON |
| < | 003C | LESS-THAN SIGN |
| = | 003D | EQUALS SIGN |
| > | 003E | GREATER-THAN SIGN |
| ? | 003F | QUESTION MARK |
| @ | 0040 | COMMERCIAL AT |
| A | 0041 | LATIN CAPITAL LETTER A |
| B | 0042 | LATIN CAPITAL LETTER B |
| C | 0043 | LATIN CAPITAL LETTER C |
| D | 0044 | LATIN CAPITAL LETTER D |
| E | 0045 | LATIN CAPITAL LETTER E |
| F | 0046 | LATIN CAPITAL LETTER F |
| G | 0047 | LATIN CAPITAL LETTER G |
| H | 0048 | LATIN CAPITAL LETTER H |
| I | 0049 | LATIN CAPITAL LETTER I |
| J | 004A | LATIN CAPITAL LETTER J |
| K | 004B | LATIN CAPITAL LETTER K |
| L | 004C | LATIN CAPITAL LETTER L |
| M | 004D | LATIN CAPITAL LETTER M |
| N | 004E | LATIN CAPITAL LETTER N |
| O | 004F | LATIN CAPITAL LETTER O |
| P | 0050 | LATIN CAPITAL LETTER P |
| Q | 0051 | LATIN CAPITAL LETTER Q |
| R | 0052 | LATIN CAPITAL LETTER R |
| S | 0053 | LATIN CAPITAL LETTER S |
| T | 0054 | LATIN CAPITAL LETTER T |
| U | 0055 | LATIN CAPITAL LETTER U |
| V | 0056 | LATIN CAPITAL LETTER V |
| W | 0057 | LATIN CAPITAL LETTER W |
| X | 0058 | LATIN CAPITAL LETTER X |
| Y | 0059 | LATIN CAPITAL LETTER Y |
| Z | 005A | LATIN CAPITAL LETTER Z |
| [ | 005B | LEFT SQUARE BRACKET |
| \ | 005C | REVERSE SOLIDUS |
| ] | 005D | RIGHT SQUARE BRACKET |
| ^ | 005E | CIRCUMFLEX ACCENT |
| \_ | 005F | LOW LINE |
| ` | 0060 | GRAVE ACCENT |
| a | 0061 | LATIN SMALL LETTER A |
| b | 0062 | LATIN SMALL LETTER B |
| c | 0063 | LATIN SMALL LETTER C |
| d | 0064 | LATIN SMALL LETTER D |
| e | 0065 | LATIN SMALL LETTER E |
| f | 0066 | LATIN SMALL LETTER F |
| g | 0067 | LATIN SMALL LETTER G |
| h | 0068 | LATIN SMALL LETTER H |
| i | 0069 | LATIN SMALL LETTER I |
| j | 006A | LATIN SMALL LETTER J |
| k | 006B | LATIN SMALL LETTER K |
| l | 006C | LATIN SMALL LETTER L |
| m | 006D | LATIN SMALL LETTER M |
| n | 006E | LATIN SMALL LETTER N |
| o | 006F | LATIN SMALL LETTER O |
| p | 0070 | LATIN SMALL LETTER P |
| q | 0071 | LATIN SMALL LETTER Q |
| r | 0072 | LATIN SMALL LETTER R |
| s | 0073 | LATIN SMALL LETTER S |
| t | 0074 | LATIN SMALL LETTER T |
| u | 0075 | LATIN SMALL LETTER U |
| v | 0076 | LATIN SMALL LETTER V |
| w | 0077 | LATIN SMALL LETTER W |
| x | 0078 | LATIN SMALL LETTER X |
| y | 0079 | LATIN SMALL LETTER Y |
| z | 007A | LATIN SMALL LETTER Z |
| { | 007B | LEFT CURLY BRACKET |
| | | 007C | VERTICAL LINE |
| } | 007D | RIGHT CURLY BRACKET |
| ~ | 007E | TILDE |
|  | 00A0 | NO-BREAK SPACE |
| ¡ | 00A1 | INVERTED EXCLAMATION MARK |
| ¢ | 00A2 | CENT SIGN |
| £ | 00A3 | POUND SIGN |
| ¤ | 00A4 | CURRENCY SIGN |
| ¥ | 00A5 | YEN SIGN |
| ¦ | 00A6 | BROKEN BAR |
| § | 00A7 | SECTION SIGN |
| ¨ | 00A8 | DIAERESIS |
| © | 00A9 | COPYRIGHT SIGN |
| ª | 00AA | FEMININE ORDINAL INDICATOR |
| « | 00AB | LEFT-POINTING DOUBLE ANGLE QUOTATION MARK |
| ¬ | 00AC | NOT SIGN |
|  | 00AD | SOFT HYPHEN |
| ® | 00AE | REGISTERED SIGN |
| ¯ | 00AF | MACRON |
| ° | 00B0 | DEGREE SIGN |
| ± | 00B1 | PLUS-MINUS SIGN |
| ² | 00B2 | SUPERSCRIPT TWO |
| ³ | 00B3 | SUPERSCRIPT THREE |
| ´ | 00B4 | ACUTE ACCENT |
| µ | 00B5 | MICRO SIGN |
| ¶ | 00B6 | PILCROW SIGN |
| · | 00B7 | MIDDLE DOT |
| ¸ | 00B8 | CEDILLA |
| ¹ | 00B9 | SUPERSCRIPT ONE |
| º | 00BA | MASCULINE ORDINAL INDICATOR |
| » | 00BB | RIGHT-POINTING DOUBLE ANGLE QUOTATION MARK |
| ¼ | 00BC | VULGAR FRACTION ONE QUARTER |
| ½ | 00BD | VULGAR FRACTION ONE HALF |
| ¾ | 00BE | VULGAR FRACTION THREE QUARTERS |
| ¿ | 00BF | INVERTED QUESTION MARK |
| À | 00C0 | LATIN CAPITAL LETTER A WITH GRAVE |
| Á | 00C1 | LATIN CAPITAL LETTER A WITH ACUTE |
| Â | 00C2 | LATIN CAPITAL LETTER A WITH CIRCUMFLEX |
| Ã | 00C3 | LATIN CAPITAL LETTER A WITH TILDE |
| Ä | 00C4 | LATIN CAPITAL LETTER A WITH DIAERESIS |
| Å | 00C5 | LATIN CAPITAL LETTER A WITH RING ABOVE |
| Æ | 00C6 | LATIN CAPITAL LETTER AE |
| Ç | 00C7 | LATIN CAPITAL LETTER C WITH CEDILLA |
| È | 00C8 | LATIN CAPITAL LETTER E WITH GRAVE |
| É | 00C9 | LATIN CAPITAL LETTER E WITH ACUTE |
| Ê | 00CA | LATIN CAPITAL LETTER E WITH CIRCUMFLEX |
| Ë | 00CB | LATIN CAPITAL LETTER E WITH DIAERESIS |
| Ì | 00CC | LATIN CAPITAL LETTER I WITH GRAVE |
| Í | 00CD | LATIN CAPITAL LETTER I WITH ACUTE |
| Î | 00CE | LATIN CAPITAL LETTER I WITH CIRCUMFLEX |
| Ï | 00CF | LATIN CAPITAL LETTER I WITH DIAERESIS |
| Ð | 00D0 | LATIN CAPITAL LETTER ETH |
| Ñ | 00D1 | LATIN CAPITAL LETTER N WITH TILDE |
| Ò | 00D2 | LATIN CAPITAL LETTER O WITH GRAVE |
| Ó | 00D3 | LATIN CAPITAL LETTER O WITH ACUTE |
| Ô | 00D4 | LATIN CAPITAL LETTER O WITH CIRCUMFLEX |
| Õ | 00D5 | LATIN CAPITAL LETTER O WITH TILDE |
| Ö | 00D6 | LATIN CAPITAL LETTER O WITH DIAERESIS |
| × | 00D7 | MULTIPLICATION SIGN |
| Ø | 00D8 | LATIN CAPITAL LETTER O WITH STROKE |
| Ù | 00D9 | LATIN CAPITAL LETTER U WITH GRAVE |
| Ú | 00DA | LATIN CAPITAL LETTER U WITH ACUTE |
| Û | 00DB | LATIN CAPITAL LETTER U WITH CIRCUMFLEX |
| Ü | 00DC | LATIN CAPITAL LETTER U WITH DIAERESIS |
| Ý | 00DD | LATIN CAPITAL LETTER Y WITH ACUTE |
| Þ | 00DE | LATIN CAPITAL LETTER THORN |
| ß | 00DF | LATIN SMALL LETTER SHARP S |
| à | 00E0 | LATIN SMALL LETTER A WITH GRAVE |
| á | 00E1 | LATIN SMALL LETTER A WITH ACUTE |
| â | 00E2 | LATIN SMALL LETTER A WITH CIRCUMFLEX |
| ã | 00E3 | LATIN SMALL LETTER A WITH TILDE |
| ä | 00E4 | LATIN SMALL LETTER A WITH DIAERESIS |
| å | 00E5 | LATIN SMALL LETTER A WITH RING ABOVE |
| æ | 00E6 | LATIN SMALL LETTER AE |
| ç | 00E7 | LATIN SMALL LETTER C WITH CEDILLA |
| è | 00E8 | LATIN SMALL LETTER E WITH GRAVE |
| é | 00E9 | LATIN SMALL LETTER E WITH ACUTE |
| ê | 00EA | LATIN SMALL LETTER E WITH CIRCUMFLEX |
| ë | 00EB | LATIN SMALL LETTER E WITH DIAERESIS |
| ì | 00EC | LATIN SMALL LETTER I WITH GRAVE |
| í | 00ED | LATIN SMALL LETTER I WITH ACUTE |
| î | 00EE | LATIN SMALL LETTER I WITH CIRCUMFLEX |
| ï | 00EF | LATIN SMALL LETTER I WITH DIAERESIS |
| ð | 00F0 | LATIN SMALL LETTER ETH |
| ñ | 00F1 | LATIN SMALL LETTER N WITH TILDE |
| ò | 00F2 | LATIN SMALL LETTER O WITH GRAVE |
| ó | 00F3 | LATIN SMALL LETTER O WITH ACUTE |
| ô | 00F4 | LATIN SMALL LETTER O WITH CIRCUMFLEX |
| õ | 00F5 | LATIN SMALL LETTER O WITH TILDE |
| ö | 00F6 | LATIN SMALL LETTER O WITH DIAERESIS |
| ÷ | 00F7 | DIVISION SIGN |
| ø | 00F8 | LATIN SMALL LETTER O WITH STROKE |
| ù | 00F9 | LATIN SMALL LETTER U WITH GRAVE |
| ú | 00FA | LATIN SMALL LETTER U WITH ACUTE |
| û | 00FB | LATIN SMALL LETTER U WITH CIRCUMFLEX |
| ü | 00FC | LATIN SMALL LETTER U WITH DIAERESIS |
| ý | 00FD | LATIN SMALL LETTER Y WITH ACUTE |
| þ | 00FE | LATIN SMALL LETTER THORN |
| ÿ | 00FF | LATIN SMALL LETTER Y WITH DIAERESIS |
| Ā | 0100 | LATIN CAPITAL LETTER A WITH MACRON |
| ā | 0101 | LATIN SMALL LETTER A WITH MACRON |
| Ă | 0102 | LATIN CAPITAL LETTER A WITH BREVE |
| ă | 0103 | LATIN SMALL LETTER A WITH BREVE |
| Ą | 0104 | LATIN CAPITAL LETTER A WITH OGONEK |
| ą | 0105 | LATIN SMALL LETTER A WITH OGONEK |
| Ć | 0106 | LATIN CAPITAL LETTER C WITH ACUTE |
| ć | 0107 | LATIN SMALL LETTER C WITH ACUTE |
| Ĉ | 0108 | LATIN CAPITAL LETTER C WITH CIRCUMFLEX |
| ĉ | 0109 | LATIN SMALL LETTER C WITH CIRCUMFLEX |
| Ċ | 010A | LATIN CAPITAL LETTER C WITH DOT ABOVE |
| ċ | 010B | LATIN SMALL LETTER C WITH DOT ABOVE |
| Č | 010C | LATIN CAPITAL LETTER C WITH CARON |
| č | 010D | LATIN SMALL LETTER C WITH CARON |
| Ď | 010E | LATIN CAPITAL LETTER D WITH CARON |
| ď | 010F | LATIN SMALL LETTER D WITH CARON |
| Đ | 0110 | LATIN CAPITAL LETTER D WITH STROKE |
| đ | 0111 | LATIN SMALL LETTER D WITH STROKE |
| Ē | 0112 | LATIN CAPITAL LETTER E WITH MACRON |
| ē | 0113 | LATIN SMALL LETTER E WITH MACRON |
| Ĕ | 0114 | LATIN CAPITAL LETTER E WITH BREVE |
| ĕ | 0115 | LATIN SMALL LETTER E WITH BREVE |
| Ė | 0116 | LATIN CAPITAL LETTER E WITH DOT ABOVE |
| ė | 0117 | LATIN SMALL LETTER E WITH DOT ABOVE |
| Ę | 0118 | LATIN CAPITAL LETTER E WITH OGONEK |
| ę | 0119 | LATIN SMALL LETTER E WITH OGONEK |
| Ě | 011A | LATIN CAPITAL LETTER E WITH CARON |
| ě | 011B | LATIN SMALL LETTER E WITH CARON |
| Ĝ | 011C | LATIN CAPITAL LETTER G WITH CIRCUMFLEX |
| ĝ | 011D | LATIN SMALL LETTER G WITH CIRCUMFLEX |
| Ğ | 011E | LATIN CAPITAL LETTER G WITH BREVE |
| ğ | 011F | LATIN SMALL LETTER G WITH BREVE |
| Ġ | 0120 | LATIN CAPITAL LETTER G WITH DOT ABOVE |
| ġ | 0121 | LATIN SMALL LETTER G WITH DOT ABOVE |
| Ģ | 0122 | LATIN CAPITAL LETTER G WITH CEDILLA |
| ģ | 0123 | LATIN SMALL LETTER G WITH CEDILLA |
| Ĥ | 0124 | LATIN CAPITAL LETTER H WITH CIRCUMFLEX |
| ĥ | 0125 | LATIN SMALL LETTER H WITH CIRCUMFLEX |
| Ħ | 0126 | LATIN CAPITAL LETTER H WITH STROKE |
| ħ | 0127 | LATIN SMALL LETTER H WITH STROKE |
| Ĩ | 0128 | LATIN CAPITAL LETTER I WITH TILDE |
| ĩ | 0129 | LATIN SMALL LETTER I WITH TILDE |
| Ī | 012A | LATIN CAPITAL LETTER I WITH MACRON |
| ī | 012B | LATIN SMALL LETTER I WITH MACRON |
| Ĭ | 012C | LATIN CAPITAL LETTER I WITH BREVE |
| ĭ | 012D | LATIN SMALL LETTER I WITH BREVE |
| Į | 012E | LATIN CAPITAL LETTER I WITH OGONEK |
| į | 012F | LATIN SMALL LETTER I WITH OGONEK |
| İ | 0130 | LATIN CAPITAL LETTER I WITH DOT ABOVE |
| ı | 0131 | LATIN SMALL LETTER DOTLESS I |
| Ĳ | 0132 | LATIN CAPITAL LIGATURE IJ |
| ĳ | 0133 | LATIN SMALL LIGATURE IJ |
| Ĵ | 0134 | LATIN CAPITAL LETTER J WITH CIRCUMFLEX |
| ĵ | 0135 | LATIN SMALL LETTER J WITH CIRCUMFLEX |
| Ķ | 0136 | LATIN CAPITAL LETTER K WITH CEDILLA |
| ķ | 0137 | LATIN SMALL LETTER K WITH CEDILLA |
| ĸ | 0138 | LATIN SMALL LETTER KRA |
| Ĺ | 0139 | LATIN CAPITAL LETTER L WITH ACUTE |
| ĺ | 013A | LATIN SMALL LETTER L WITH ACUTE |
| Ļ | 013B | LATIN CAPITAL LETTER L WITH CEDILLA |
| ļ | 013C | LATIN SMALL LETTER L WITH CEDILLA |
| Ľ | 013D | LATIN CAPITAL LETTER L WITH CARON |
| ľ | 013E | LATIN SMALL LETTER L WITH CARON |
| Ŀ | 013F | LATIN CAPITAL LETTER L WITH MIDDLE DOT |
| ŀ | 0140 | LATIN SMALL LETTER L WITH MIDDLE DOT |
| Ł | 0141 | LATIN CAPITAL LETTER L WITH STROKE |
| ł | 0142 | LATIN SMALL LETTER L WITH STROKE |
| Ń | 0143 | LATIN CAPITAL LETTER N WITH ACUTE |
| ń | 0144 | LATIN SMALL LETTER N WITH ACUTE |
| Ņ | 0145 | LATIN CAPITAL LETTER N WITH CEDILLA |
| ņ | 0146 | LATIN SMALL LETTER N WITH CEDILLA |
| Ň | 0147 | LATIN CAPITAL LETTER N WITH CARON |
| ň | 0148 | LATIN SMALL LETTER N WITH CARON |
| ŉ | 0149 | LATIN SMALL LETTER N PRECEDED BY APOSTROPHE |
| Ŋ | 014A | LATIN CAPITAL LETTER ENG |
| ŋ | 014B | LATIN SMALL LETTER ENG |
| Ō | 014C | LATIN CAPITAL LETTER O WITH MACRON |
| ō | 014D | LATIN SMALL LETTER O WITH MACRON |
| Ŏ | 014E | LATIN CAPITAL LETTER O WITH BREVE |
| ŏ | 014F | LATIN SMALL LETTER O WITH BREVE |
| Ő | 0150 | LATIN CAPITAL LETTER O WITH DOUBLE ACUTE |
| ő | 0151 | LATIN SMALL LETTER O WITH DOUBLE ACUTE |
| Œ | 0152 | LATIN CAPITAL LIGATURE OE |
| œ | 0153 | LATIN SMALL LIGATURE OE |
| Ŕ | 0154 | LATIN CAPITAL LETTER R WITH ACUTE |
| ŕ | 0155 | LATIN SMALL LETTER R WITH ACUTE |
| Ŗ | 0156 | LATIN CAPITAL LETTER R WITH CEDILLA |
| ŗ | 0157 | LATIN SMALL LETTER R WITH CEDILLA |
| Ř | 0158 | LATIN CAPITAL LETTER R WITH CARON |
| ř | 0159 | LATIN SMALL LETTER R WITH CARON |
| Ś | 015A | LATIN CAPITAL LETTER S WITH ACUTE |
| ś | 015B | LATIN SMALL LETTER S WITH ACUTE |
| Ŝ | 015C | LATIN CAPITAL LETTER S WITH CIRCUMFLEX |
| ŝ | 015D | LATIN SMALL LETTER S WITH CIRCUMFLEX |
| Ş | 015E | LATIN CAPITAL LETTER S WITH CEDILLA |
| ş | 015F | LATIN SMALL LETTER S WITH CEDILLA |
| Š | 0160 | LATIN CAPITAL LETTER S WITH CARON |
| š | 0161 | LATIN SMALL LETTER S WITH CARON |
| Ţ | 0162 | LATIN CAPITAL LETTER T WITH CEDILLA |
| ţ | 0163 | LATIN SMALL LETTER T WITH CEDILLA |
| Ť | 0164 | LATIN CAPITAL LETTER T WITH CARON |
| ť | 0165 | LATIN SMALL LETTER T WITH CARON |
| Ŧ | 0166 | LATIN CAPITAL LETTER T WITH STROKE |
| ŧ | 0167 | LATIN SMALL LETTER T WITH STROKE |
| Ũ | 0168 | LATIN CAPITAL LETTER U WITH TILDE |
| ũ | 0169 | LATIN SMALL LETTER U WITH TILDE |
| Ū | 016A | LATIN CAPITAL LETTER U WITH MACRON |
| ū | 016B | LATIN SMALL LETTER U WITH MACRON |
| Ŭ | 016C | LATIN CAPITAL LETTER U WITH BREVE |
| ŭ | 016D | LATIN SMALL LETTER U WITH BREVE |
| Ů | 016E | LATIN CAPITAL LETTER U WITH RING ABOVE |
| ů | 016F | LATIN SMALL LETTER U WITH RING ABOVE |
| Ű | 0170 | LATIN CAPITAL LETTER U WITH DOUBLE ACUTE |
| ű | 0171 | LATIN SMALL LETTER U WITH DOUBLE ACUTE |
| Ų | 0172 | LATIN CAPITAL LETTER U WITH OGONEK |
| ų | 0173 | LATIN SMALL LETTER U WITH OGONEK |
| Ŵ | 0174 | LATIN CAPITAL LETTER W WITH CIRCUMFLEX |
| ŵ | 0175 | LATIN SMALL LETTER W WITH CIRCUMFLEX |
| Ŷ | 0176 | LATIN CAPITAL LETTER Y WITH CIRCUMFLEX |
| ŷ | 0177 | LATIN SMALL LETTER Y WITH CIRCUMFLEX |
| Ÿ | 0178 | LATIN CAPITAL LETTER Y WITH DIAERESIS |
| Ź | 0179 | LATIN CAPITAL LETTER Z WITH ACUTE |
| ź | 017A | LATIN SMALL LETTER Z WITH ACUTE |
| Ż | 017B | LATIN CAPITAL LETTER Z WITH DOT ABOVE |
| ż | 017C | LATIN SMALL LETTER Z WITH DOT ABOVE |
| Ž | 017D | LATIN CAPITAL LETTER Z WITH CARON |
| ž | 017E | LATIN SMALL LETTER Z WITH CARON |
| ſ | 017F | LATIN SMALL LETTER LONG S |
| ƀ | 0180 | LATIN SMALL LETTER B WITH STROKE |
| Ɓ | 0181 | LATIN CAPITAL LETTER B WITH HOOK |
| Ƃ | 0182 | LATIN CAPITAL LETTER B WITH TOPBAR |
| ƃ | 0183 | LATIN SMALL LETTER B WITH TOPBAR |
| Ƅ | 0184 | LATIN CAPITAL LETTER TONE SIX |
| ƅ | 0185 | LATIN SMALL LETTER TONE SIX |
| Ɔ | 0186 | LATIN CAPITAL LETTER OPEN O |
| Ƈ | 0187 | LATIN CAPITAL LETTER C WITH HOOK |
| ƈ | 0188 | LATIN SMALL LETTER C WITH HOOK |
| Ɖ | 0189 | LATIN CAPITAL LETTER AFRICAN D |
| Ɗ | 018A | LATIN CAPITAL LETTER D WITH HOOK |
| Ƌ | 018B | LATIN CAPITAL LETTER D WITH TOPBAR |
| ƌ | 018C | LATIN SMALL LETTER D WITH TOPBAR |
| ƍ | 018D | LATIN SMALL LETTER TURNED DELTA |
| Ǝ | 018E | LATIN CAPITAL LETTER REVERSED E |
| Ə | 018F | LATIN CAPITAL LETTER SCHWA |
| Ɛ | 0190 | LATIN CAPITAL LETTER OPEN E |
| Ƒ | 0191 | LATIN CAPITAL LETTER F WITH HOOK |
| ƒ | 0192 | LATIN SMALL LETTER F WITH HOOK |
| Ɠ | 0193 | LATIN CAPITAL LETTER G WITH HOOK |
| Ɣ | 0194 | LATIN CAPITAL LETTER GAMMA |
| ƕ | 0195 | LATIN SMALL LETTER HV |
| Ɩ | 0196 | LATIN CAPITAL LETTER IOTA |
| Ɨ | 0197 | LATIN CAPITAL LETTER I WITH STROKE |
| Ƙ | 0198 | LATIN CAPITAL LETTER K WITH HOOK |
| ƙ | 0199 | LATIN SMALL LETTER K WITH HOOK |
| ƚ | 019A | LATIN SMALL LETTER L WITH BAR |
| ƛ | 019B | LATIN SMALL LETTER LAMBDA WITH STROKE |
| Ɯ | 019C | LATIN CAPITAL LETTER TURNED M |
| Ɲ | 019D | LATIN CAPITAL LETTER N WITH LEFT HOOK |
| ƞ | 019E | LATIN SMALL LETTER N WITH LONG RIGHT LEG |
| Ɵ | 019F | LATIN CAPITAL LETTER O WITH MIDDLE TILDE |
| Ơ | 01A0 | LATIN CAPITAL LETTER O WITH HORN |
| ơ | 01A1 | LATIN SMALL LETTER O WITH HORN |
| Ƣ | 01A2 | LATIN CAPITAL LETTER OI |
| ƣ | 01A3 | LATIN SMALL LETTER OI |
| Ƥ | 01A4 | LATIN CAPITAL LETTER P WITH HOOK |
| ƥ | 01A5 | LATIN SMALL LETTER P WITH HOOK |
| Ʀ | 01A6 | LATIN LETTER YR |
| Ƨ | 01A7 | LATIN CAPITAL LETTER TONE TWO |
| ƨ | 01A8 | LATIN SMALL LETTER TONE TWO |
| Ʃ | 01A9 | LATIN CAPITAL LETTER ESH |
| ƪ | 01AA | LATIN LETTER REVERSED ESH LOOP |
| ƫ | 01AB | LATIN SMALL LETTER T WITH PALATAL HOOK |
| Ƭ | 01AC | LATIN CAPITAL LETTER T WITH HOOK |
| ƭ | 01AD | LATIN SMALL LETTER T WITH HOOK |
| Ʈ | 01AE | LATIN CAPITAL LETTER T WITH RETROFLEX HOOK |
| Ư | 01AF | LATIN CAPITAL LETTER U WITH HORN |
| ư | 01B0 | LATIN SMALL LETTER U WITH HORN |
| Ʊ | 01B1 | LATIN CAPITAL LETTER UPSILON |
| Ʋ | 01B2 | LATIN CAPITAL LETTER V WITH HOOK |
| Ƴ | 01B3 | LATIN CAPITAL LETTER Y WITH HOOK |
| ƴ | 01B4 | LATIN SMALL LETTER Y WITH HOOK |
| Ƶ | 01B5 | LATIN CAPITAL LETTER Z WITH STROKE |
| ƶ | 01B6 | LATIN SMALL LETTER Z WITH STROKE |
| Ʒ | 01B7 | LATIN CAPITAL LETTER EZH |
| Ƹ | 01B8 | LATIN CAPITAL LETTER EZH REVERSED |
| ƹ | 01B9 | LATIN SMALL LETTER EZH REVERSED |
| ƺ | 01BA | LATIN SMALL LETTER EZH WITH TAIL |
| ƻ | 01BB | LATIN LETTER TWO WITH STROKE |
| Ƽ | 01BC | LATIN CAPITAL LETTER TONE FIVE |
| ƽ | 01BD | LATIN SMALL LETTER TONE FIVE |
| ƾ | 01BE | LATIN LETTER INVERTED GLOTTAL STOP WITH STROKE |
| ƿ | 01BF | LATIN LETTER WYNN |
| ǀ | 01C0 | LATIN LETTER DENTAL CLICK |
| ǁ | 01C1 | LATIN LETTER LATERAL CLICK |
| ǂ | 01C2 | LATIN LETTER ALVEOLAR CLICK |
| ǃ | 01C3 | LATIN LETTER RETROFLEX CLICK |
| Ǆ | 01C4 | LATIN CAPITAL LETTER DZ WITH CARON |
| ǅ | 01C5 | LATIN CAPITAL LETTER D WITH SMALL LETTER Z WITH CARON |
| ǆ | 01C6 | LATIN SMALL LETTER DZ WITH CARON |
| Ǉ | 01C7 | LATIN CAPITAL LETTER LJ |
| ǈ | 01C8 | LATIN CAPITAL LETTER L WITH SMALL LETTER J |
| ǉ | 01C9 | LATIN SMALL LETTER LJ |
| Ǌ | 01CA | LATIN CAPITAL LETTER NJ |
| ǋ | 01CB | LATIN CAPITAL LETTER N WITH SMALL LETTER J |
| ǌ | 01CC | LATIN SMALL LETTER NJ |
| Ǎ | 01CD | LATIN CAPITAL LETTER A WITH CARON |
| ǎ | 01CE | LATIN SMALL LETTER A WITH CARON |
| Ǐ | 01CF | LATIN CAPITAL LETTER I WITH CARON |
| ǐ | 01D0 | LATIN SMALL LETTER I WITH CARON |
| Ǒ | 01D1 | LATIN CAPITAL LETTER O WITH CARON |
| ǒ | 01D2 | LATIN SMALL LETTER O WITH CARON |
| Ǔ | 01D3 | LATIN CAPITAL LETTER U WITH CARON |
| ǔ | 01D4 | LATIN SMALL LETTER U WITH CARON |
| Ǖ | 01D5 | LATIN CAPITAL LETTER U WITH DIAERESIS AND MACRON |
| ǖ | 01D6 | LATIN SMALL LETTER U WITH DIAERESIS AND MACRON |
| Ǘ | 01D7 | LATIN CAPITAL LETTER U WITH DIAERESIS AND ACUTE |
| ǘ | 01D8 | LATIN SMALL LETTER U WITH DIAERESIS AND ACUTE |
| Ǚ | 01D9 | LATIN CAPITAL LETTER U WITH DIAERESIS AND CARON |
| ǚ | 01DA | LATIN SMALL LETTER U WITH DIAERESIS AND CARON |
| Ǜ | 01DB | LATIN CAPITAL LETTER U WITH DIAERESIS AND GRAVE |
| ǜ | 01DC | LATIN SMALL LETTER U WITH DIAERESIS AND GRAVE |
| ǝ | 01DD | LATIN SMALL LETTER TURNED E |
| Ǟ | 01DE | LATIN CAPITAL LETTER A WITH DIAERESIS AND MACRON |
| ǟ | 01DF | LATIN SMALL LETTER A WITH DIAERESIS AND MACRON |
| Ǡ | 01E0 | LATIN CAPITAL LETTER A WITH DOT ABOVE AND MACRON |
| ǡ | 01E1 | LATIN SMALL LETTER A WITH DOT ABOVE AND MACRON |
| Ǣ | 01E2 | LATIN CAPITAL LETTER AE WITH MACRON |
| ǣ | 01E3 | LATIN SMALL LETTER AE WITH MACRON |
| Ǥ | 01E4 | LATIN CAPITAL LETTER G WITH STROKE |
| ǥ | 01E5 | LATIN SMALL LETTER G WITH STROKE |
| Ǧ | 01E6 | LATIN CAPITAL LETTER G WITH CARON |
| ǧ | 01E7 | LATIN SMALL LETTER G WITH CARON |
| Ǩ | 01E8 | LATIN CAPITAL LETTER K WITH CARON |
| ǩ | 01E9 | LATIN SMALL LETTER K WITH CARON |
| Ǫ | 01EA | LATIN CAPITAL LETTER O WITH OGONEK |
| ǫ | 01EB | LATIN SMALL LETTER O WITH OGONEK |
| Ǭ | 01EC | LATIN CAPITAL LETTER O WITH OGONEK AND MACRON |
| ǭ | 01ED | LATIN SMALL LETTER O WITH OGONEK AND MACRON |
| Ǯ | 01EE | LATIN CAPITAL LETTER EZH WITH CARON |
| ǯ | 01EF | LATIN SMALL LETTER EZH WITH CARON |
| ǰ | 01F0 | LATIN SMALL LETTER J WITH CARON |
| Ǳ | 01F1 | LATIN CAPITAL LETTER DZ |
| ǲ | 01F2 | LATIN CAPITAL LETTER D WITH SMALL LETTER Z |
| ǳ | 01F3 | LATIN SMALL LETTER DZ |
| Ǵ | 01F4 | LATIN CAPITAL LETTER G WITH ACUTE |
| ǵ | 01F5 | LATIN SMALL LETTER G WITH ACUTE |
| Ƕ | 01F6 | LATIN CAPITAL LETTER HWAIR |
| Ƿ | 01F7 | LATIN CAPITAL LETTER WYNN |
| Ǹ | 01F8 | LATIN CAPITAL LETTER N WITH GRAVE |
| ǹ | 01F9 | LATIN SMALL LETTER N WITH GRAVE |
| Ǻ | 01FA | LATIN CAPITAL LETTER A WITH RING ABOVE AND ACUTE |
| ǻ | 01FB | LATIN SMALL LETTER A WITH RING ABOVE AND ACUTE |
| Ǽ | 01FC | LATIN CAPITAL LETTER AE WITH ACUTE |
| ǽ | 01FD | LATIN SMALL LETTER AE WITH ACUTE |
| Ǿ | 01FE | LATIN CAPITAL LETTER O WITH STROKE AND ACUTE |
| ǿ | 01FF | LATIN SMALL LETTER O WITH STROKE AND ACUTE |
| Ȁ | 0200 | LATIN CAPITAL LETTER A WITH DOUBLE GRAVE |
| ȁ | 0201 | LATIN SMALL LETTER A WITH DOUBLE GRAVE |
| Ȃ | 0202 | LATIN CAPITAL LETTER A WITH INVERTED BREVE |
| ȃ | 0203 | LATIN SMALL LETTER A WITH INVERTED BREVE |
| Ȅ | 0204 | LATIN CAPITAL LETTER E WITH DOUBLE GRAVE |
| ȅ | 0205 | LATIN SMALL LETTER E WITH DOUBLE GRAVE |
| Ȇ | 0206 | LATIN CAPITAL LETTER E WITH INVERTED BREVE |
| ȇ | 0207 | LATIN SMALL LETTER E WITH INVERTED BREVE |
| Ȉ | 0208 | LATIN CAPITAL LETTER I WITH DOUBLE GRAVE |
| ȉ | 0209 | LATIN SMALL LETTER I WITH DOUBLE GRAVE |
| Ȋ | 020A | LATIN CAPITAL LETTER I WITH INVERTED BREVE |
| ȋ | 020B | LATIN SMALL LETTER I WITH INVERTED BREVE |
| Ȍ | 020C | LATIN CAPITAL LETTER O WITH DOUBLE GRAVE |
| ȍ | 020D | LATIN SMALL LETTER O WITH DOUBLE GRAVE |
| Ȏ | 020E | LATIN CAPITAL LETTER O WITH INVERTED BREVE |
| ȏ | 020F | LATIN SMALL LETTER O WITH INVERTED BREVE |
| Ȑ | 0210 | LATIN CAPITAL LETTER R WITH DOUBLE GRAVE |
| ȑ | 0211 | LATIN SMALL LETTER R WITH DOUBLE GRAVE |
| Ȓ | 0212 | LATIN CAPITAL LETTER R WITH INVERTED BREVE |
| ȓ | 0213 | LATIN SMALL LETTER R WITH INVERTED BREVE |
| Ȕ | 0214 | LATIN CAPITAL LETTER U WITH DOUBLE GRAVE |
| ȕ | 0215 | LATIN SMALL LETTER U WITH DOUBLE GRAVE |
| Ȗ | 0216 | LATIN CAPITAL LETTER U WITH INVERTED BREVE |
| ȗ | 0217 | LATIN SMALL LETTER U WITH INVERTED BREVE |
| Ș | 0218 | LATIN CAPITAL LETTER S WITH COMMA BELOW |
| ș | 0219 | LATIN SMALL LETTER S WITH COMMA BELOW |
| Ț | 021A | LATIN CAPITAL LETTER T WITH COMMA BELOW |
| ț | 021B | LATIN SMALL LETTER T WITH COMMA BELOW |
| Ȝ | 021C | LATIN CAPITAL LETTER YOGH |
| ȝ | 021D | LATIN SMALL LETTER YOGH |
| Ȟ | 021E | LATIN CAPITAL LETTER H WITH CARON |
| ȟ | 021F | LATIN SMALL LETTER H WITH CARON |
| Ƞ | 0220 | LATIN CAPITAL LETTER N WITH LONG RIGHT LEG |
| ȡ | 0221 | LATIN SMALL LETTER D WITH CURL |
| Ȣ | 0222 | LATIN CAPITAL LETTER OU |
| ȣ | 0223 | LATIN SMALL LETTER OU |
| Ȥ | 0224 | LATIN CAPITAL LETTER Z WITH HOOK |
| ȥ | 0225 | LATIN SMALL LETTER Z WITH HOOK |
| Ȧ | 0226 | LATIN CAPITAL LETTER A WITH DOT ABOVE |
| ȧ | 0227 | LATIN SMALL LETTER A WITH DOT ABOVE |
| Ȩ | 0228 | LATIN CAPITAL LETTER E WITH CEDILLA |
| ȩ | 0229 | LATIN SMALL LETTER E WITH CEDILLA |
| Ȫ | 022A | LATIN CAPITAL LETTER O WITH DIAERESIS AND MACRON |
| ȫ | 022B | LATIN SMALL LETTER O WITH DIAERESIS AND MACRON |
| Ȭ | 022C | LATIN CAPITAL LETTER O WITH TILDE AND MACRON |
| ȭ | 022D | LATIN SMALL LETTER O WITH TILDE AND MACRON |
| Ȯ | 022E | LATIN CAPITAL LETTER O WITH DOT ABOVE |
| ȯ | 022F | LATIN SMALL LETTER O WITH DOT ABOVE |
| Ȱ | 0230 | LATIN CAPITAL LETTER O WITH DOT ABOVE AND MACRON |
| ȱ | 0231 | LATIN SMALL LETTER O WITH DOT ABOVE AND MACRON |
| Ȳ | 0232 | LATIN CAPITAL LETTER Y WITH MACRON |
| ȳ | 0233 | LATIN SMALL LETTER Y WITH MACRON |
| ȴ | 0234 | LATIN SMALL LETTER L WITH CURL |
| ȵ | 0235 | LATIN SMALL LETTER N WITH CURL |
| ȶ | 0236 | LATIN SMALL LETTER T WITH CURL |
| ȷ | 0237 | LATIN SMALL LETTER DOTLESS J |
| ȸ | 0238 | LATIN SMALL LETTER DB DIGRAPH |
| ȹ | 0239 | LATIN SMALL LETTER QP DIGRAPH |
| Ⱥ | 023A | LATIN CAPITAL LETTER A WITH STROKE |
| Ȼ | 023B | LATIN CAPITAL LETTER C WITH STROKE |
| ȼ | 023C | LATIN SMALL LETTER C WITH STROKE |
| Ƚ | 023D | LATIN CAPITAL LETTER L WITH BAR |
| Ⱦ | 023E | LATIN CAPITAL LETTER T WITH DIAGONAL STROKE |
| ȿ | 023F | LATIN SMALL LETTER S WITH SWASH TAIL |
| ɀ | 0240 | LATIN SMALL LETTER Z WITH SWASH TAIL |
| Ɂ | 0241 | LATIN CAPITAL LETTER GLOTTAL STOP |
| ɂ | 0242 | LATIN SMALL LETTER GLOTTAL STOP |
| Ƀ | 0243 | LATIN CAPITAL LETTER B WITH STROKE |
| Ʉ | 0244 | LATIN CAPITAL LETTER U BAR |
| Ʌ | 0245 | LATIN CAPITAL LETTER TURNED V |
| Ɇ | 0246 | LATIN CAPITAL LETTER E WITH STROKE |
| ɇ | 0247 | LATIN SMALL LETTER E WITH STROKE |
| Ɉ | 0248 | LATIN CAPITAL LETTER J WITH STROKE |
| ɉ | 0249 | LATIN SMALL LETTER J WITH STROKE |
| Ɋ | 024A | LATIN CAPITAL LETTER SMALL Q WITH HOOK TAIL |
| ɋ | 024B | LATIN SMALL LETTER Q WITH HOOK TAIL |
| Ɍ | 024C | LATIN CAPITAL LETTER R WITH STROKE |
| ɍ | 024D | LATIN SMALL LETTER R WITH STROKE |
| Ɏ | 024E | LATIN CAPITAL LETTER Y WITH STROKE |
| ɏ | 024F | LATIN SMALL LETTER Y WITH STROKE |
| ʰ | 02B0 | MODIFIER LETTER SMALL H |
| ʱ | 02B1 | MODIFIER LETTER SMALL H WITH HOOK |
| ʲ | 02B2 | MODIFIER LETTER SMALL J |
| ʳ | 02B3 | MODIFIER LETTER SMALL R |
| ʴ | 02B4 | MODIFIER LETTER SMALL TURNED R |
| ʵ | 02B5 | MODIFIER LETTER SMALL TURNED R WITH HOOK |
| ʶ | 02B6 | MODIFIER LETTER SMALL CAPITAL INVERTED R |
| ʷ | 02B7 | MODIFIER LETTER SMALL W |
| ʸ | 02B8 | MODIFIER LETTER SMALL Y |
| ʹ | 02B9 | MODIFIER LETTER PRIME |
| ʺ | 02BA | MODIFIER LETTER DOUBLE PRIME |
| ʻ | 02BB | MODIFIER LETTER TURNED COMMA |
| ʼ | 02BC | MODIFIER LETTER APOSTROPHE |
| ʽ | 02BD | MODIFIER LETTER REVERSED COMMA |
| ʾ | 02BE | MODIFIER LETTER RIGHT HALF RING |
| ʿ | 02BF | MODIFIER LETTER LEFT HALF RING |
| ˀ | 02C0 | MODIFIER LETTER GLOTTAL STOP |
| ˁ | 02C1 | MODIFIER LETTER REVERSED GLOTTAL STOP |
| ˂ | 02C2 | MODIFIER LETTER LEFT ARROWHEAD |
| ˃ | 02C3 | MODIFIER LETTER RIGHT ARROWHEAD |
| ˄ | 02C4 | MODIFIER LETTER UP ARROWHEAD |
| ˅ | 02C5 | MODIFIER LETTER DOWN ARROWHEAD |
| ˆ | 02C6 | MODIFIER LETTER CIRCUMFLEX ACCENT |
| ˇ | 02C7 | CARON |
| ˈ | 02C8 | MODIFIER LETTER VERTICAL LINE |
| ˉ | 02C9 | MODIFIER LETTER MACRON |
| ˊ | 02CA | MODIFIER LETTER ACUTE ACCENT |
| ˋ | 02CB | MODIFIER LETTER GRAVE ACCENT |
| ˌ | 02CC | MODIFIER LETTER LOW VERTICAL LINE |
| ˍ | 02CD | MODIFIER LETTER LOW MACRON |
| ˎ | 02CE | MODIFIER LETTER LOW GRAVE ACCENT |
| ˏ | 02CF | MODIFIER LETTER LOW ACUTE ACCENT |
| ː | 02D0 | MODIFIER LETTER TRIANGULAR COLON |
| ˑ | 02D1 | MODIFIER LETTER HALF TRIANGULAR COLON |
| ˒ | 02D2 | MODIFIER LETTER CENTRED RIGHT HALF RING |
| ˓ | 02D3 | MODIFIER LETTER CENTRED LEFT HALF RING |
| ˔ | 02D4 | MODIFIER LETTER UP TACK |
| ˕ | 02D5 | MODIFIER LETTER DOWN TACK |
| ˖ | 02D6 | MODIFIER LETTER PLUS SIGN |
| ˗ | 02D7 | MODIFIER LETTER MINUS SIGN |
| ˘ | 02D8 | BREVE |
| ˙ | 02D9 | DOT ABOVE |
| ˚ | 02DA | RING ABOVE |
| ˛ | 02DB | OGONEK |
| ˜ | 02DC | SMALL TILDE |
| ˝ | 02DD | DOUBLE ACUTE ACCENT |
| ˞ | 02DE | MODIFIER LETTER RHOTIC HOOK |
| ˟ | 02DF | MODIFIER LETTER CROSS ACCENT |
| ˠ | 02E0 | MODIFIER LETTER SMALL GAMMA |
| ˡ | 02E1 | MODIFIER LETTER SMALL L |
| ˢ | 02E2 | MODIFIER LETTER SMALL S |
| ˣ | 02E3 | MODIFIER LETTER SMALL X |
| ˤ | 02E4 | MODIFIER LETTER SMALL REVERSED GLOTTAL STOP |
| ˥ | 02E5 | MODIFIER LETTER EXTRA-HIGH TONE BAR |
| ˦ | 02E6 | MODIFIER LETTER HIGH TONE BAR |
| ˧ | 02E7 | MODIFIER LETTER MID TONE BAR |
| ˨ | 02E8 | MODIFIER LETTER LOW TONE BAR |
| ˩ | 02E9 | MODIFIER LETTER EXTRA-LOW TONE BAR |
| ˪ | 02EA | MODIFIER LETTER YIN DEPARTING TONE MARK |
| ˫ | 02EB | MODIFIER LETTER YANG DEPARTING TONE MARK |
| ˬ | 02EC | MODIFIER LETTER VOICING |
| ˭ | 02ED | MODIFIER LETTER UNASPIRATED |
| ˮ | 02EE | MODIFIER LETTER DOUBLE APOSTROPHE |
| ˯ | 02EF | MODIFIER LETTER LOW DOWN ARROWHEAD |
| ˰ | 02F0 | MODIFIER LETTER LOW UP ARROWHEAD |
| ˱ | 02F1 | MODIFIER LETTER LOW LEFT ARROWHEAD |
| ˲ | 02F2 | MODIFIER LETTER LOW RIGHT ARROWHEAD |
| ˳ | 02F3 | MODIFIER LETTER LOW RING |
| ˴ | 02F4 | MODIFIER LETTER MIDDLE GRAVE ACCENT |
| ˵ | 02F5 | MODIFIER LETTER MIDDLE DOUBLE GRAVE ACCENT |
| ˶ | 02F6 | MODIFIER LETTER MIDDLE DOUBLE ACUTE ACCENT |
| ˷ | 02F7 | MODIFIER LETTER LOW TILDE |
| ˸ | 02F8 | MODIFIER LETTER RAISED COLON |
| ˹ | 02F9 | MODIFIER LETTER BEGIN HIGH TONE |
| ˺ | 02FA | MODIFIER LETTER END HIGH TONE |
| ˻ | 02FB | MODIFIER LETTER BEGIN LOW TONE |
| ˼ | 02FC | MODIFIER LETTER END LOW TONE |
| ˽ | 02FD | MODIFIER LETTER SHELF |
| ˾ | 02FE | MODIFIER LETTER OPEN SHELF |
| ˿ | 02FF | MODIFIER LETTER LOW LEFT ARROW |
| ò | 0300 | GRAVE ACCENT |
| ó | 0301 | ACUTE ACCENT |
| ô | 0302 | CIRCUMFLEX ACCENT |
| õ | 0303 | TILDE |
| ō | 0304 | MACRON |
| o̅ | 0305 | OVERLINE |
| ŏ | 0306 | BREVE |
| ȯ | 0307 | DOT ABOVE |
| ö | 0308 | DIAERESIS |
| ỏ | 0309 | HOOK ABOVE |
| o̊ | 030A | RING ABOVE |
| ő | 030B | DOUBLE ACUTE ACCENT |
| ǒ | 030C | CARON |
| o̍ | 030D | VERTICAL LINE ABOVE |
| o̎ | 030E | DOUBLE VERTICAL LINE ABOVE |
| ȍ | 030F | DOUBLE GRAVE ACCENT |
| o̐ | 0310 | CANDRABINDU |
| ȏ | 0311 | INVERTED BREVE |
| o̒ | 0312 | TURNED COMMA ABOVE |
| o̓ | 0313 | COMMA ABOVE |
| o̔ | 0314 | REVERSED COMMA ABOVE |
| o̕ | 0315 | COMMA ABOVE RIGHT |
| o̖ | 0316 | GRAVE ACCENT BELOW |
| o̗ | 0317 | ACUTE ACCENT BELOW |
| o̘ | 0318 | LEFT TACK BELOW |
| o̙ | 0319 | RIGHT TACK BELOW |
| o̚ | 031A | LEFT ANGLE ABOVE |
| ơ | 031B | HORN |
| o̜ | 031C | LEFT HALF RING BELOW |
| o̝ | 031D | UP TACK BELOW |
| o̞ | 031E | DOWN TACK BELOW |
| o̟ | 031F | PLUS SIGN BELOW |
| o̠ | 0320 | MINUS SIGN BELOW |
| o̡ | 0321 | PALATALIZED HOOK BELOW |
| o̢ | 0322 | RETROFLEX HOOK BELOW |
| ọ | 0323 | DOT BELOW |
| o̤ | 0324 | DIAERESIS BELOW |
| o̥ | 0325 | RING BELOW |
| o̦ | 0326 | COMMA BELOW |
| o̧ | 0327 | CEDILLA |
| ǫ | 0328 | OGONEK |
| o̩ | 0329 | VERTICAL LINE BELOW |
| o̪ | 032A | BRIDGE BELOW |
| o̫ | 032B | INVERTED DOUBLE ARCH BELOW |
| o̬ | 032C | CARON BELOW |
| o̭ | 032D | CIRCUMFLEX ACCENT BELOW |
| o̮ | 032E | BREVE BELOW |
| o̯ | 032F | INVERTED BREVE BELOW |
| o̰ | 0330 | TILDE BELOW |
| o̱ | 0331 | MACRON BELOW |
| o̲ | 0332 | LOW LINE |
| o̳ | 0333 | DOUBLE LOW LINE |
| o̴ | 0334 | TILDE OVERLAY |
| o̵ | 0335 | SHORT STROKE OVERLAY |
| o̶ | 0336 | LONG STROKE OVERLAY |
| o̷ | 0337 | SHORT SOLIDUS OVERLAY |
| o̸ | 0338 | LONG SOLIDUS OVERLAY |
| o̹ | 0339 | RIGHT HALF RING BELOW |
| o̺ | 033A | INVERTED BRIDGE BELOW |
| o̻ | 033B | SQUARE BELOW |
| o̼ | 033C | SEAGULL BELOW |
| o̽ | 033D | X ABOVE |
| o̾ | 033E | VERTICAL TILDE |
| o̿ | 033F | DOUBLE OVERLINE |
| ò | 0340 | GRAVE TONE MARK |
| ó | 0341 | ACUTE TONE MARK |
| o͂ | 0342 | GREEK PERISPOMENI (combined with theta) |
| o̓ | 0343 | GREEK KORONIS (combined with theta) |
| ö́ | 0344 | GREEK DIALYTIKA TONOS (combined with theta) |
| oͅ | 0345 | GREEK YPOGEGRAMMENI (combined with theta) |
| o͆ | 0346 | BRIDGE ABOVE |
| o͇ | 0347 | EQUALS SIGN BELOW |
| o͈ | 0348 | DOUBLE VERTICAL LINE BELOW |
| o͉ | 0349 | LEFT ANGLE BELOW |
| o͊ | 034A | NOT TILDE ABOVE |
| o͋ | 034B | HOMOTHETIC ABOVE |
| o͌ | 034C | ALMOST EQUAL TO ABOVE |
| o͍ | 034D | LEFT RIGHT ARROW BELOW |
| o͎ | 034E | UPWARDS ARROW BELOW |
| o͏ | 034F | GRAPHEME JOINER |
| o͐ | 0350 | RIGHT ARROWHEAD ABOVE |
| o͑ | 0351 | LEFT HALF RING ABOVE |
| o͒ | 0352 | FERMATA |
| o͓ | 0353 | X BELOW |
| o͔ | 0354 | LEFT ARROWHEAD BELOW |
| o͕ | 0355 | RIGHT ARROWHEAD BELOW |
| o͖ | 0356 | RIGHT ARROWHEAD AND UP ARROWHEAD BELOW |
| o͗ | 0357 | RIGHT HALF RING ABOVE |
| o͘ | 0358 | DOT ABOVE RIGHT |
| o͙ | 0359 | ASTERISK BELOW |
| o͚ | 035A | DOUBLE RING BELOW |
| o͛ | 035B | ZIGZAG ABOVE |
| ͜o | 035C | DOUBLE BREVE BELOW |
| ͝o | 035D | DOUBLE BREVE |
| ͞o | 035E | DOUBLE MACRON |
| ͟o | 035F | DOUBLE MACRON BELOW |
| ͠o | 0360 | DOUBLE TILDE |
| ͡o | 0361 | DOUBLE INVERTED BREVE |
| ͢o | 0362 | DOUBLE RIGHTWARDS ARROW BELOW |
| oͣ | 0363 | LATIN SMALL LETTER A |
| oͤ | 0364 | LATIN SMALL LETTER E |
| oͥ | 0365 | LATIN SMALL LETTER I |
| oͦ | 0366 | LATIN SMALL LETTER O |
| oͧ | 0367 | LATIN SMALL LETTER U |
| oͨ | 0368 | LATIN SMALL LETTER C |
| oͩ | 0369 | LATIN SMALL LETTER D |
| oͪ | 036A | LATIN SMALL LETTER H |
| oͫ | 036B | LATIN SMALL LETTER M |
| oͬ | 036C | LATIN SMALL LETTER R |
| oͭ | 036D | LATIN SMALL LETTER T |
| oͮ | 036E | LATIN SMALL LETTER V |
| oͯ | 036F | LATIN SMALL LETTER X |
| Ͱ | 0370 | GREEK CAPITAL LETTER HETA |
| ͱ | 0371 | GREEK SMALL LETTER HETA |
| Ͳ | 0372 | GREEK CAPITAL LETTER ARCHAIC SAMPI |
| ͳ | 0373 | GREEK SMALL LETTER ARCHAIC SAMPI |
| ʹ | 0374 | GREEK NUMERAL SIGN |
| ͵ | 0375 | GREEK LOWER NUMERAL SIGN |
| Ͷ | 0376 | GREEK CAPITAL LETTER PAMPHYLIAN DIGAMMA |
| ͷ | 0377 | GREEK SMALL LETTER PAMPHYLIAN DIGAMMA |
| ͺ | 037A | GREEK YPOGEGRAMMENI |
| ͻ | 037B | GREEK SMALL REVERSED LUNATE SIGMA SYMBOL |
| ͼ | 037C | GREEK SMALL DOTTED LUNATE SIGMA SYMBOL |
| ͽ | 037D | GREEK SMALL REVERSED DOTTED LUNATE SIGMA SYMBOL |
| ; | 037E | GREEK QUESTION MARK |
| ΄ | 0384 | GREEK TONOS |
| ΅ | 0385 | GREEK DIALYTIKA TONOS |
| Ά | 0386 | GREEK CAPITAL LETTER ALPHA WITH TONOS |
| · | 0387 | GREEK ANO TELEIA |
| Έ | 0388 | GREEK CAPITAL LETTER EPSILON WITH TONOS |
| Ή | 0389 | GREEK CAPITAL LETTER ETA WITH TONOS |
| Ί | 038A | GREEK CAPITAL LETTER IOTA WITH TONOS |
| Ό | 038C | GREEK CAPITAL LETTER OMICRON WITH TONOS |
| Ύ | 038E | GREEK CAPITAL LETTER UPSILON WITH TONOS |
| Ώ | 038F | GREEK CAPITAL LETTER OMEGA WITH TONOS |
| ΐ | 0390 | GREEK SMALL LETTER IOTA WITH DIALYTIKA AND TONOS |
| Α | 0391 | GREEK CAPITAL LETTER ALPHA |
| Β | 0392 | GREEK CAPITAL LETTER BETA |
| Γ | 0393 | GREEK CAPITAL LETTER GAMMA |
| Δ | 0394 | GREEK CAPITAL LETTER DELTA |
| Ε | 0395 | GREEK CAPITAL LETTER EPSILON |
| Ζ | 0396 | GREEK CAPITAL LETTER ZETA |
| Η | 0397 | GREEK CAPITAL LETTER ETA |
| Θ | 0398 | GREEK CAPITAL LETTER THETA |
| Ι | 0399 | GREEK CAPITAL LETTER IOTA |
| Κ | 039A | GREEK CAPITAL LETTER KAPPA |
| Λ | 039B | GREEK CAPITAL LETTER LAMBDA |
| Μ | 039C | GREEK CAPITAL LETTER MU |
| Ν | 039D | GREEK CAPITAL LETTER NU |
| Ξ | 039E | GREEK CAPITAL LETTER XI |
| Ο | 039F | GREEK CAPITAL LETTER OMICRON |
| Π | 03A0 | GREEK CAPITAL LETTER PI |
| Ρ | 03A1 | GREEK CAPITAL LETTER RHO |
| Σ | 03A3 | GREEK CAPITAL LETTER SIGMA |
| Τ | 03A4 | GREEK CAPITAL LETTER TAU |
| Υ | 03A5 | GREEK CAPITAL LETTER UPSILON |
| Φ | 03A6 | GREEK CAPITAL LETTER PHI |
| Χ | 03A7 | GREEK CAPITAL LETTER CHI |
| Ψ | 03A8 | GREEK CAPITAL LETTER PSI |
| Ω | 03A9 | GREEK CAPITAL LETTER OMEGA |
| Ϊ | 03AA | GREEK CAPITAL LETTER IOTA WITH DIALYTIKA |
| Ϋ | 03AB | GREEK CAPITAL LETTER UPSILON WITH DIALYTIKA |
| ά | 03AC | GREEK SMALL LETTER ALPHA WITH TONOS |
| έ | 03AD | GREEK SMALL LETTER EPSILON WITH TONOS |
| ή | 03AE | GREEK SMALL LETTER ETA WITH TONOS |
| ί | 03AF | GREEK SMALL LETTER IOTA WITH TONOS |
| ΰ | 03B0 | GREEK SMALL LETTER UPSILON WITH DIALYTIKA AND TONOS |
| α | 03B1 | GREEK SMALL LETTER ALPHA |
| β | 03B2 | GREEK SMALL LETTER BETA |
| γ | 03B3 | GREEK SMALL LETTER GAMMA |
| δ | 03B4 | GREEK SMALL LETTER DELTA |
| ε | 03B5 | GREEK SMALL LETTER EPSILON |
| ζ | 03B6 | GREEK SMALL LETTER ZETA |
| η | 03B7 | GREEK SMALL LETTER ETA |
| θ | 03B8 | GREEK SMALL LETTER THETA |
| ι | 03B9 | GREEK SMALL LETTER IOTA |
| κ | 03BA | GREEK SMALL LETTER KAPPA |
| λ | 03BB | GREEK SMALL LETTER LAMBDA |
| μ | 03BC | GREEK SMALL LETTER MU |
| ν | 03BD | GREEK SMALL LETTER NU |
| ξ | 03BE | GREEK SMALL LETTER XI |
| ο | 03BF | GREEK SMALL LETTER OMICRON |
| π | 03C0 | GREEK SMALL LETTER PI |
| ρ | 03C1 | GREEK SMALL LETTER RHO |
| ς | 03C2 | GREEK SMALL LETTER FINAL SIGMA |
| σ | 03C3 | GREEK SMALL LETTER SIGMA |
| τ | 03C4 | GREEK SMALL LETTER TAU |
| υ | 03C5 | GREEK SMALL LETTER UPSILON |
| φ | 03C6 | GREEK SMALL LETTER PHI |
| χ | 03C7 | GREEK SMALL LETTER CHI |
| ψ | 03C8 | GREEK SMALL LETTER PSI |
| ω | 03C9 | GREEK SMALL LETTER OMEGA |
| ϊ | 03CA | GREEK SMALL LETTER IOTA WITH DIALYTIKA |
| ϋ | 03CB | GREEK SMALL LETTER UPSILON WITH DIALYTIKA |
| ό | 03CC | GREEK SMALL LETTER OMICRON WITH TONOS |
| ύ | 03CD | GREEK SMALL LETTER UPSILON WITH TONOS |
| ώ | 03CE | GREEK SMALL LETTER OMEGA WITH TONOS |
| Ϗ | 03CF | GREEK CAPITAL KAI SYMBOL |
| ϐ | 03D0 | GREEK BETA SYMBOL |
| ϑ | 03D1 | GREEK THETA SYMBOL |
| ϒ | 03D2 | GREEK UPSILON WITH HOOK SYMBOL |
| ϓ | 03D3 | GREEK UPSILON WITH ACUTE AND HOOK SYMBOL |
| ϔ | 03D4 | GREEK UPSILON WITH DIAERESIS AND HOOK SYMBOL |
| ϕ | 03D5 | GREEK PHI SYMBOL |
| ϖ | 03D6 | GREEK PI SYMBOL |
| ϗ | 03D7 | GREEK KAI SYMBOL |
| Ϙ | 03D8 | GREEK LETTER ARCHAIC KOPPA |
| ϙ | 03D9 | GREEK SMALL LETTER ARCHAIC KOPPA |
| Ϛ | 03DA | GREEK LETTER STIGMA |
| ϛ | 03DB | GREEK SMALL LETTER STIGMA |
| Ϝ | 03DC | GREEK LETTER DIGAMMA |
| ϝ | 03DD | GREEK SMALL LETTER DIGAMMA |
| Ϟ | 03DE | GREEK LETTER KOPPA |
| ϟ | 03DF | GREEK SMALL LETTER KOPPA |
| Ϡ | 03E0 | GREEK LETTER SAMPI |
| ϡ | 03E1 | GREEK SMALL LETTER SAMPI |
| Ϣ | 03E2 | COPTIC CAPITAL LETTER SHEI |
| ϣ | 03E3 | COPTIC SMALL LETTER SHEI |
| Ϥ | 03E4 | COPTIC CAPITAL LETTER FEI |
| ϥ | 03E5 | COPTIC SMALL LETTER FEI |
| Ϧ | 03E6 | COPTIC CAPITAL LETTER KHEI |
| ϧ | 03E7 | COPTIC SMALL LETTER KHEI |
| Ϩ | 03E8 | COPTIC CAPITAL LETTER HORI |
| ϩ | 03E9 | COPTIC SMALL LETTER HORI |
| Ϫ | 03EA | COPTIC CAPITAL LETTER GANGIA |
| ϫ | 03EB | COPTIC SMALL LETTER GANGIA |
| Ϭ | 03EC | COPTIC CAPITAL LETTER SHIMA |
| ϭ | 03ED | COPTIC SMALL LETTER SHIMA |
| Ϯ | 03EE | COPTIC CAPITAL LETTER DEI |
| ϯ | 03EF | COPTIC SMALL LETTER DEI |
| ϰ | 03F0 | GREEK KAPPA SYMBOL |
| ϱ | 03F1 | GREEK RHO SYMBOL |
| ϲ | 03F2 | GREEK LUNATE SIGMA SYMBOL |
| ϳ | 03F3 | GREEK LETTER YOT |
| ϴ | 03F4 | GREEK CAPITAL THETA SYMBOL |
| ϵ | 03F5 | GREEK LUNATE EPSILON SYMBOL |
| ϶ | 03F6 | GREEK REVERSED LUNATE EPSILON SYMBOL |
| Ϸ | 03F7 | GREEK CAPITAL LETTER SHO |
| ϸ | 03F8 | GREEK SMALL LETTER SHO |
| Ϲ | 03F9 | GREEK CAPITAL LUNATE SIGMA SYMBOL |
| Ϻ | 03FA | GREEK CAPITAL LETTER SAN |
| ϻ | 03FB | GREEK SMALL LETTER SAN |
| ϼ | 03FC | GREEK RHO WITH STROKE SYMBOL |
| Ͻ | 03FD | GREEK CAPITAL REVERSED LUNATE SIGMA SYMBOL |
| Ͼ | 03FE | GREEK CAPITAL DOTTED LUNATE SIGMA SYMBOL |
| Ͽ | 03FF | GREEK CAPITAL REVERSED DOTTED LUNATE SIGMA SYMBOL |
| Ѐ | 0400 | CYRILLIC CAPITAL LETTER IE WITH GRAVE |
| Ё | 0401 | CYRILLIC CAPITAL LETTER IO |
| Ђ | 0402 | CYRILLIC CAPITAL LETTER DJE |
| Ѓ | 0403 | CYRILLIC CAPITAL LETTER GJE |
| Є | 0404 | CYRILLIC CAPITAL LETTER UKRAINIAN IE |
| Ѕ | 0405 | CYRILLIC CAPITAL LETTER DZE |
| І | 0406 | CYRILLIC CAPITAL LETTER BYELORUSSIAN-UKRAINIAN I |
| Ї | 0407 | CYRILLIC CAPITAL LETTER YI |
| Ј | 0408 | CYRILLIC CAPITAL LETTER JE |
| Љ | 0409 | CYRILLIC CAPITAL LETTER LJE |
| Њ | 040A | CYRILLIC CAPITAL LETTER NJE |
| Ћ | 040B | CYRILLIC CAPITAL LETTER TSHE |
| Ќ | 040C | CYRILLIC CAPITAL LETTER KJE |
| Ѝ | 040D | CYRILLIC CAPITAL LETTER I WITH GRAVE |
| Ў | 040E | CYRILLIC CAPITAL LETTER SHORT U |
| Џ | 040F | CYRILLIC CAPITAL LETTER DZHE |
| А | 0410 | CYRILLIC CAPITAL LETTER A |
| Б | 0411 | CYRILLIC CAPITAL LETTER BE |
| В | 0412 | CYRILLIC CAPITAL LETTER VE |
| Г | 0413 | CYRILLIC CAPITAL LETTER GHE |
| Д | 0414 | CYRILLIC CAPITAL LETTER DE |
| Е | 0415 | CYRILLIC CAPITAL LETTER IE |
| Ж | 0416 | CYRILLIC CAPITAL LETTER ZHE |
| З | 0417 | CYRILLIC CAPITAL LETTER ZE |
| И | 0418 | CYRILLIC CAPITAL LETTER I |
| Й | 0419 | CYRILLIC CAPITAL LETTER SHORT I |
| К | 041A | CYRILLIC CAPITAL LETTER KA |
| Л | 041B | CYRILLIC CAPITAL LETTER EL |
| М | 041C | CYRILLIC CAPITAL LETTER EM |
| Н | 041D | CYRILLIC CAPITAL LETTER EN |
| О | 041E | CYRILLIC CAPITAL LETTER O |
| П | 041F | CYRILLIC CAPITAL LETTER PE |
| Р | 0420 | CYRILLIC CAPITAL LETTER ER |
| С | 0421 | CYRILLIC CAPITAL LETTER ES |
| Т | 0422 | CYRILLIC CAPITAL LETTER TE |
| У | 0423 | CYRILLIC CAPITAL LETTER U |
| Ф | 0424 | CYRILLIC CAPITAL LETTER EF |
| Х | 0425 | CYRILLIC CAPITAL LETTER HA |
| Ц | 0426 | CYRILLIC CAPITAL LETTER TSE |
| Ч | 0427 | CYRILLIC CAPITAL LETTER CHE |
| Ш | 0428 | CYRILLIC CAPITAL LETTER SHA |
| Щ | 0429 | CYRILLIC CAPITAL LETTER SHCHA |
| Ъ | 042A | CYRILLIC CAPITAL LETTER HARD SIGN |
| Ы | 042B | CYRILLIC CAPITAL LETTER YERU |
| Ь | 042C | CYRILLIC CAPITAL LETTER SOFT SIGN |
| Э | 042D | CYRILLIC CAPITAL LETTER E |
| Ю | 042E | CYRILLIC CAPITAL LETTER YU |
| Я | 042F | CYRILLIC CAPITAL LETTER YA |
| а | 0430 | CYRILLIC SMALL LETTER A |
| б | 0431 | CYRILLIC SMALL LETTER BE |
| в | 0432 | CYRILLIC SMALL LETTER VE |
| г | 0433 | CYRILLIC SMALL LETTER GHE |
| д | 0434 | CYRILLIC SMALL LETTER DE |
| е | 0435 | CYRILLIC SMALL LETTER IE |
| ж | 0436 | CYRILLIC SMALL LETTER ZHE |
| з | 0437 | CYRILLIC SMALL LETTER ZE |
| и | 0438 | CYRILLIC SMALL LETTER I |
| й | 0439 | CYRILLIC SMALL LETTER SHORT I |
| к | 043A | CYRILLIC SMALL LETTER KA |
| л | 043B | CYRILLIC SMALL LETTER EL |
| м | 043C | CYRILLIC SMALL LETTER EM |
| н | 043D | CYRILLIC SMALL LETTER EN |
| о | 043E | CYRILLIC SMALL LETTER O |
| п | 043F | CYRILLIC SMALL LETTER PE |
| р | 0440 | CYRILLIC SMALL LETTER ER |
| с | 0441 | CYRILLIC SMALL LETTER ES |
| т | 0442 | CYRILLIC SMALL LETTER TE |
| у | 0443 | CYRILLIC SMALL LETTER U |
| ф | 0444 | CYRILLIC SMALL LETTER EF |
| х | 0445 | CYRILLIC SMALL LETTER HA |
| ц | 0446 | CYRILLIC SMALL LETTER TSE |
| ч | 0447 | CYRILLIC SMALL LETTER CHE |
| ш | 0448 | CYRILLIC SMALL LETTER SHA |
| щ | 0449 | CYRILLIC SMALL LETTER SHCHA |
| ъ | 044A | CYRILLIC SMALL LETTER HARD SIGN |
| ы | 044B | CYRILLIC SMALL LETTER YERU |
| ь | 044C | CYRILLIC SMALL LETTER SOFT SIGN |
| э | 044D | CYRILLIC SMALL LETTER E |
| ю | 044E | CYRILLIC SMALL LETTER YU |
| я | 044F | CYRILLIC SMALL LETTER YA |
| ѐ | 0450 | CYRILLIC SMALL LETTER IE WITH GRAVE |
| ё | 0451 | CYRILLIC SMALL LETTER IO |
| ђ | 0452 | CYRILLIC SMALL LETTER DJE |
| ѓ | 0453 | CYRILLIC SMALL LETTER GJE |
| є | 0454 | CYRILLIC SMALL LETTER UKRAINIAN IE |
| ѕ | 0455 | CYRILLIC SMALL LETTER DZE |
| і | 0456 | CYRILLIC SMALL LETTER BYELORUSSIAN-UKRAINIAN I |
| ї | 0457 | CYRILLIC SMALL LETTER YI |
| ј | 0458 | CYRILLIC SMALL LETTER JE |
| љ | 0459 | CYRILLIC SMALL LETTER LJE |
| њ | 045A | CYRILLIC SMALL LETTER NJE |
| ћ | 045B | CYRILLIC SMALL LETTER TSHE |
| ќ | 045C | CYRILLIC SMALL LETTER KJE |
| ѝ | 045D | CYRILLIC SMALL LETTER I WITH GRAVE |
| ў | 045E | CYRILLIC SMALL LETTER SHORT U |
| џ | 045F | CYRILLIC SMALL LETTER DZHE |
| Ѡ | 0460 | CYRILLIC CAPITAL LETTER OMEGA |
| ѡ | 0461 | CYRILLIC SMALL LETTER OMEGA |
| Ѣ | 0462 | CYRILLIC CAPITAL LETTER YAT |
| ѣ | 0463 | CYRILLIC SMALL LETTER YAT |
| Ѥ | 0464 | CYRILLIC CAPITAL LETTER IOTIFIED E |
| ѥ | 0465 | CYRILLIC SMALL LETTER IOTIFIED E |
| Ѧ | 0466 | CYRILLIC CAPITAL LETTER LITTLE YUS |
| ѧ | 0467 | CYRILLIC SMALL LETTER LITTLE YUS |
| Ѩ | 0468 | CYRILLIC CAPITAL LETTER IOTIFIED LITTLE YUS |
| ѩ | 0469 | CYRILLIC SMALL LETTER IOTIFIED LITTLE YUS |
| Ѫ | 046A | CYRILLIC CAPITAL LETTER BIG YUS |
| ѫ | 046B | CYRILLIC SMALL LETTER BIG YUS |
| Ѭ | 046C | CYRILLIC CAPITAL LETTER IOTIFIED BIG YUS |
| ѭ | 046D | CYRILLIC SMALL LETTER IOTIFIED BIG YUS |
| Ѯ | 046E | CYRILLIC CAPITAL LETTER KSI |
| ѯ | 046F | CYRILLIC SMALL LETTER KSI |
| Ѱ | 0470 | CYRILLIC CAPITAL LETTER PSI |
| ѱ | 0471 | CYRILLIC SMALL LETTER PSI |
| Ѳ | 0472 | CYRILLIC CAPITAL LETTER FITA |
| ѳ | 0473 | CYRILLIC SMALL LETTER FITA |
| Ѵ | 0474 | CYRILLIC CAPITAL LETTER IZHITSA |
| ѵ | 0475 | CYRILLIC SMALL LETTER IZHITSA |
| Ѷ | 0476 | CYRILLIC CAPITAL LETTER IZHITSA WITH DOUBLE GRAVE ACCENT |
| ѷ | 0477 | CYRILLIC SMALL LETTER IZHITSA WITH DOUBLE GRAVE ACCENT |
| Ѹ | 0478 | CYRILLIC CAPITAL LETTER UK |
| ѹ | 0479 | CYRILLIC SMALL LETTER UK |
| Ѻ | 047A | CYRILLIC CAPITAL LETTER ROUND OMEGA |
| ѻ | 047B | CYRILLIC SMALL LETTER ROUND OMEGA |
| Ѽ | 047C | CYRILLIC CAPITAL LETTER OMEGA WITH TITLO |
| ѽ | 047D | CYRILLIC SMALL LETTER OMEGA WITH TITLO |
| Ѿ | 047E | CYRILLIC CAPITAL LETTER OT |
| ѿ | 047F | CYRILLIC SMALL LETTER OT |
| Ҁ | 0480 | CYRILLIC CAPITAL LETTER KOPPA |
| ҁ | 0481 | CYRILLIC SMALL LETTER KOPPA |
| ҂ | 0482 | CYRILLIC THOUSANDS SIGN |
| о҃ | 0483 | COMBINING CYRILLIC TITLO (combined with о) |
| о҄ | 0484 | COMBINING CYRILLIC PALATALIZATION (combined with о) |
| о҅ | 0485 | COMBINING CYRILLIC DASIA PNEUMATA (combined with о) |
| о҆ | 0486 | COMBINING CYRILLIC PSILI PNEUMATA (combined with о) |
| о҇ | 0487 | COMBINING CYRILLIC POKRYTIE (combined with о) |
| о҈ | 0488 | COMBINING CYRILLIC HUNDRED THOUSANDS SIGN (combined with о) |
| о҉ | 0489 | COMBINING CYRILLIC MILLIONS SIGN (combined with о) |
| Ҋ | 048A | CYRILLIC CAPITAL LETTER SHORT I WITH TAIL |
| ҋ | 048B | CYRILLIC SMALL LETTER SHORT I WITH TAIL |
| Ҍ | 048C | CYRILLIC CAPITAL LETTER SEMISOFT SIGN |
| ҍ | 048D | CYRILLIC SMALL LETTER SEMISOFT SIGN |
| Ҏ | 048E | CYRILLIC CAPITAL LETTER ER WITH TICK |
| ҏ | 048F | CYRILLIC SMALL LETTER ER WITH TICK |
| Ґ | 0490 | CYRILLIC CAPITAL LETTER GHE WITH UPTURN |
| ґ | 0491 | CYRILLIC SMALL LETTER GHE WITH UPTURN |
| Ғ | 0492 | CYRILLIC CAPITAL LETTER GHE WITH STROKE |
| ғ | 0493 | CYRILLIC SMALL LETTER GHE WITH STROKE |
| Ҕ | 0494 | CYRILLIC CAPITAL LETTER GHE WITH MIDDLE HOOK |
| ҕ | 0495 | CYRILLIC SMALL LETTER GHE WITH MIDDLE HOOK |
| Җ | 0496 | CYRILLIC CAPITAL LETTER ZHE WITH DESCENDER |
| җ | 0497 | CYRILLIC SMALL LETTER ZHE WITH DESCENDER |
| Ҙ | 0498 | CYRILLIC CAPITAL LETTER ZE WITH DESCENDER |
| ҙ | 0499 | CYRILLIC SMALL LETTER ZE WITH DESCENDER |
| Қ | 049A | CYRILLIC CAPITAL LETTER KA WITH DESCENDER |
| қ | 049B | CYRILLIC SMALL LETTER KA WITH DESCENDER |
| Ҝ | 049C | CYRILLIC CAPITAL LETTER KA WITH VERTICAL STROKE |
| ҝ | 049D | CYRILLIC SMALL LETTER KA WITH VERTICAL STROKE |
| Ҟ | 049E | CYRILLIC CAPITAL LETTER KA WITH STROKE |
| ҟ | 049F | CYRILLIC SMALL LETTER KA WITH STROKE |
| Ҡ | 04A0 | CYRILLIC CAPITAL LETTER BASHKIR KA |
| ҡ | 04A1 | CYRILLIC SMALL LETTER BASHKIR KA |
| Ң | 04A2 | CYRILLIC CAPITAL LETTER EN WITH DESCENDER |
| ң | 04A3 | CYRILLIC SMALL LETTER EN WITH DESCENDER |
| Ҥ | 04A4 | CYRILLIC CAPITAL LIGATURE EN GHE |
| ҥ | 04A5 | CYRILLIC SMALL LIGATURE EN GHE |
| Ҧ | 04A6 | CYRILLIC CAPITAL LETTER PE WITH MIDDLE HOOK |
| ҧ | 04A7 | CYRILLIC SMALL LETTER PE WITH MIDDLE HOOK |
| Ҩ | 04A8 | CYRILLIC CAPITAL LETTER ABKHASIAN HA |
| ҩ | 04A9 | CYRILLIC SMALL LETTER ABKHASIAN HA |
| Ҫ | 04AA | CYRILLIC CAPITAL LETTER ES WITH DESCENDER |
| ҫ | 04AB | CYRILLIC SMALL LETTER ES WITH DESCENDER |
| Ҭ | 04AC | CYRILLIC CAPITAL LETTER TE WITH DESCENDER |
| ҭ | 04AD | CYRILLIC SMALL LETTER TE WITH DESCENDER |
| Ү | 04AE | CYRILLIC CAPITAL LETTER STRAIGHT U |
| ү | 04AF | CYRILLIC SMALL LETTER STRAIGHT U |
| Ұ | 04B0 | CYRILLIC CAPITAL LETTER STRAIGHT U WITH STROKE |
| ұ | 04B1 | CYRILLIC SMALL LETTER STRAIGHT U WITH STROKE |
| Ҳ | 04B2 | CYRILLIC CAPITAL LETTER HA WITH DESCENDER |
| ҳ | 04B3 | CYRILLIC SMALL LETTER HA WITH DESCENDER |
| Ҵ | 04B4 | CYRILLIC CAPITAL LIGATURE TE TSE |
| ҵ | 04B5 | CYRILLIC SMALL LIGATURE TE TSE |
| Ҷ | 04B6 | CYRILLIC CAPITAL LETTER CHE WITH DESCENDER |
| ҷ | 04B7 | CYRILLIC SMALL LETTER CHE WITH DESCENDER |
| Ҹ | 04B8 | CYRILLIC CAPITAL LETTER CHE WITH VERTICAL STROKE |
| ҹ | 04B9 | CYRILLIC SMALL LETTER CHE WITH VERTICAL STROKE |
| Һ | 04BA | CYRILLIC CAPITAL LETTER SHHA |
| һ | 04BB | CYRILLIC SMALL LETTER SHHA |
| Ҽ | 04BC | CYRILLIC CAPITAL LETTER ABKHASIAN CHE |
| ҽ | 04BD | CYRILLIC SMALL LETTER ABKHASIAN CHE |
| Ҿ | 04BE | CYRILLIC CAPITAL LETTER ABKHASIAN CHE WITH DESCENDER |
| ҿ | 04BF | CYRILLIC SMALL LETTER ABKHASIAN CHE WITH DESCENDER |
| Ӏ | 04C0 | CYRILLIC LETTER PALOCHKA |
| Ӂ | 04C1 | CYRILLIC CAPITAL LETTER ZHE WITH BREVE |
| ӂ | 04C2 | CYRILLIC SMALL LETTER ZHE WITH BREVE |
| Ӄ | 04C3 | CYRILLIC CAPITAL LETTER KA WITH HOOK |
| ӄ | 04C4 | CYRILLIC SMALL LETTER KA WITH HOOK |
| Ӆ | 04C5 | CYRILLIC CAPITAL LETTER EL WITH TAIL |
| ӆ | 04C6 | CYRILLIC SMALL LETTER EL WITH TAIL |
| Ӈ | 04C7 | CYRILLIC CAPITAL LETTER EN WITH HOOK |
| ӈ | 04C8 | CYRILLIC SMALL LETTER EN WITH HOOK |
| Ӊ | 04C9 | CYRILLIC CAPITAL LETTER EN WITH TAIL |
| ӊ | 04CA | CYRILLIC SMALL LETTER EN WITH TAIL |
| Ӌ | 04CB | CYRILLIC CAPITAL LETTER KHAKASSIAN CHE |
| ӌ | 04CC | CYRILLIC SMALL LETTER KHAKASSIAN CHE |
| Ӎ | 04CD | CYRILLIC CAPITAL LETTER EM WITH TAIL |
| ӎ | 04CE | CYRILLIC SMALL LETTER EM WITH TAIL |
| ӏ | 04CF | CYRILLIC SMALL LETTER PALOCHKA |
| Ӑ | 04D0 | CYRILLIC CAPITAL LETTER A WITH BREVE |
| ӑ | 04D1 | CYRILLIC SMALL LETTER A WITH BREVE |
| Ӓ | 04D2 | CYRILLIC CAPITAL LETTER A WITH DIAERESIS |
| ӓ | 04D3 | CYRILLIC SMALL LETTER A WITH DIAERESIS |
| Ӕ | 04D4 | CYRILLIC CAPITAL LIGATURE A IE |
| ӕ | 04D5 | CYRILLIC SMALL LIGATURE A IE |
| Ӗ | 04D6 | CYRILLIC CAPITAL LETTER IE WITH BREVE |
| ӗ | 04D7 | CYRILLIC SMALL LETTER IE WITH BREVE |
| Ә | 04D8 | CYRILLIC CAPITAL LETTER SCHWA |
| ә | 04D9 | CYRILLIC SMALL LETTER SCHWA |
| Ӛ | 04DA | CYRILLIC CAPITAL LETTER SCHWA WITH DIAERESIS |
| ӛ | 04DB | CYRILLIC SMALL LETTER SCHWA WITH DIAERESIS |
| Ӝ | 04DC | CYRILLIC CAPITAL LETTER ZHE WITH DIAERESIS |
| ӝ | 04DD | CYRILLIC SMALL LETTER ZHE WITH DIAERESIS |
| Ӟ | 04DE | CYRILLIC CAPITAL LETTER ZE WITH DIAERESIS |
| ӟ | 04DF | CYRILLIC SMALL LETTER ZE WITH DIAERESIS |
| Ӡ | 04E0 | CYRILLIC CAPITAL LETTER ABKHASIAN DZE |
| ӡ | 04E1 | CYRILLIC SMALL LETTER ABKHASIAN DZE |
| Ӣ | 04E2 | CYRILLIC CAPITAL LETTER I WITH MACRON |
| ӣ | 04E3 | CYRILLIC SMALL LETTER I WITH MACRON |
| Ӥ | 04E4 | CYRILLIC CAPITAL LETTER I WITH DIAERESIS |
| ӥ | 04E5 | CYRILLIC SMALL LETTER I WITH DIAERESIS |
| Ӧ | 04E6 | CYRILLIC CAPITAL LETTER O WITH DIAERESIS |
| ӧ | 04E7 | CYRILLIC SMALL LETTER O WITH DIAERESIS |
| Ө | 04E8 | CYRILLIC CAPITAL LETTER BARRED O |
| ө | 04E9 | CYRILLIC SMALL LETTER BARRED O |
| Ӫ | 04EA | CYRILLIC CAPITAL LETTER BARRED O WITH DIAERESIS |
| ӫ | 04EB | CYRILLIC SMALL LETTER BARRED O WITH DIAERESIS |
| Ӭ | 04EC | CYRILLIC CAPITAL LETTER E WITH DIAERESIS |
| ӭ | 04ED | CYRILLIC SMALL LETTER E WITH DIAERESIS |
| Ӯ | 04EE | CYRILLIC CAPITAL LETTER U WITH MACRON |
| ӯ | 04EF | CYRILLIC SMALL LETTER U WITH MACRON |
| Ӱ | 04F0 | CYRILLIC CAPITAL LETTER U WITH DIAERESIS |
| ӱ | 04F1 | CYRILLIC SMALL LETTER U WITH DIAERESIS |
| Ӳ | 04F2 | CYRILLIC CAPITAL LETTER U WITH DOUBLE ACUTE |
| ӳ | 04F3 | CYRILLIC SMALL LETTER U WITH DOUBLE ACUTE |
| Ӵ | 04F4 | CYRILLIC CAPITAL LETTER CHE WITH DIAERESIS |
| ӵ | 04F5 | CYRILLIC SMALL LETTER CHE WITH DIAERESIS |
| Ӷ | 04F6 | CYRILLIC CAPITAL LETTER GHE WITH DESCENDER |
| ӷ | 04F7 | CYRILLIC SMALL LETTER GHE WITH DESCENDER |
| Ӹ | 04F8 | CYRILLIC CAPITAL LETTER YERU WITH DIAERESIS |
| ӹ | 04F9 | CYRILLIC SMALL LETTER YERU WITH DIAERESIS |
| Ӻ | 04FA | CYRILLIC CAPITAL LETTER GHE WITH STROKE AND HOOK |
| ӻ | 04FB | CYRILLIC SMALL LETTER GHE WITH STROKE AND HOOK |
| Ӽ | 04FC | CYRILLIC CAPITAL LETTER HA WITH HOOK |
| ӽ | 04FD | CYRILLIC SMALL LETTER HA WITH HOOK |
| Ӿ | 04FE | CYRILLIC CAPITAL LETTER HA WITH STROKE |
| ӿ | 04FF | CYRILLIC SMALL LETTER HA WITH STROKE |
| Ԁ | 0500 | CYRILLIC CAPITAL LETTER KOMI DE |
| ԁ | 0501 | CYRILLIC SMALL LETTER KOMI DE |
| Ԃ | 0502 | CYRILLIC CAPITAL LETTER KOMI DJE |
| ԃ | 0503 | CYRILLIC SMALL LETTER KOMI DJE |
| Ԅ | 0504 | CYRILLIC CAPITAL LETTER KOMI ZJE |
| ԅ | 0505 | CYRILLIC SMALL LETTER KOMI ZJE |
| Ԇ | 0506 | CYRILLIC CAPITAL LETTER KOMI DZJE |
| ԇ | 0507 | CYRILLIC SMALL LETTER KOMI DZJE |
| Ԉ | 0508 | CYRILLIC CAPITAL LETTER KOMI LJE |
| ԉ | 0509 | CYRILLIC SMALL LETTER KOMI LJE |
| Ԋ | 050A | CYRILLIC CAPITAL LETTER KOMI NJE |
| ԋ | 050B | CYRILLIC SMALL LETTER KOMI NJE |
| Ԍ | 050C | CYRILLIC CAPITAL LETTER KOMI SJE |
| ԍ | 050D | CYRILLIC SMALL LETTER KOMI SJE |
| Ԏ | 050E | CYRILLIC CAPITAL LETTER KOMI TJE |
| ԏ | 050F | CYRILLIC SMALL LETTER KOMI TJE |
| Ԕ | 0514 | CYRILLIC CAPITAL LETTER LHA |
| ԕ | 0515 | CYRILLIC SMALL LETTER LHA |
| Ԗ | 0516 | CYRILLIC CAPITAL LETTER RHA |
| ԗ | 0517 | CYRILLIC SMALL LETTER RHA |
| Ԙ | 0518 | CYRILLIC CAPITAL LETTER YAE |
| ԙ | 0519 | CYRILLIC SMALL LETTER YAE |
| Ԛ | 051A | CYRILLIC CAPITAL LETTER QA |
| ԛ | 051B | CYRILLIC SMALL LETTER QA |
| Ԝ | 051C | CYRILLIC CAPITAL LETTER WE |
| ԝ | 051D | CYRILLIC SMALL LETTER WE |
| Ԟ | 051E | CYRILLIC CAPITAL LETTER ALEUT KA |
| ԟ | 051F | CYRILLIC SMALL LETTER ALEUT KA |
| Ԡ | 0520 | CYRILLIC CAPITAL LETTER EL WITH MIDDLE HOOK |
| ԡ | 0521 | CYRILLIC SMALL LETTER EL WITH MIDDLE HOOK |
| Ԣ | 0522 | CYRILLIC CAPITAL LETTER EN WITH MIDDLE HOOK |
| ԣ | 0523 | CYRILLIC SMALL LETTER EN WITH MIDDLE HOOK |
| Ԥ | 0524 | CYRILLIC CAPITAL LETTER PE WITH DESCENDER |
| ԥ | 0525 | CYRILLIC SMALL LETTER PE WITH DESCENDER |
| Ԧ | 0526 | CYRILLIC CAPITAL LETTER SHHA WITH DESCENDER |
| ԧ | 0527 | CYRILLIC SMALL LETTER SHHA WITH DESCENDER |
| Ԩ | 0528 |  |
| ԩ | 0529 |  |
| Ԫ | 052A |  |
| ԫ | 052B |  |
| Ԭ | 052C |  |
| ԭ | 052D |  |
| Ԯ | 052E |  |
| ԯ | 052F |  |
|  | 2000 | EN QUAD |
|  | 2001 | EM QUAD |
|  | 2002 | EN SPACE |
|  | 2003 | EM SPACE |
|  | 2004 | THREE-PER-EM SPACE |
|  | 2005 | FOUR-PER-EM SPACE |
|  | 2006 | SIX-PER-EM SPACE |
|  | 2007 | FIGURE SPACE |
|  | 2008 | PUNCTUATION SPACE |
|  | 2009 | THIN SPACE |
|  | 200A | HAIR SPACE |
| ​ | 200B | ZERO WIDTH SPACE |
| ‌ | 200C | ZERO WIDTH NON-JOINER |
| ‍ | 200D | ZERO WIDTH JOINER |
| ‎ | 200E | LEFT-TO-RIGHT MARK |
| ‏ | 200F | RIGHT-TO-LEFT MARK |
| ‐ | 2010 | HYPHEN |
| - | 2011 | NON-BREAKING HYPHEN |
| ‒ | 2012 | FIGURE DASH |
| – | 2013 | EN DASH |
| — | 2014 | EM DASH |
| ― | 2015 | HORIZONTAL BAR |
| ‖ | 2016 | DOUBLE VERTICAL LINE |
| ‗ | 2017 | DOUBLE LOW LINE |
| ‘ | 2018 | LEFT SINGLE QUOTATION MARK |
| ’ | 2019 | RIGHT SINGLE QUOTATION MARK |
| ‚ | 201A | SINGLE LOW-9 QUOTATION MARK |
| ‛ | 201B | SINGLE HIGH-REVERSED-9 QUOTATION MARK |
| “ | 201C | LEFT DOUBLE QUOTATION MARK |
| ” | 201D | RIGHT DOUBLE QUOTATION MARK |
| „ | 201E | DOUBLE LOW-9 QUOTATION MARK |
| ‟ | 201F | DOUBLE HIGH-REVERSED-9 QUOTATION MARK |
| † | 2020 | DAGGER |
| ‡ | 2021 | DOUBLE DAGGER |
| • | 2022 | BULLET |
| ‣ | 2023 | TRIANGULAR BULLET |
| ․ | 2024 | ONE DOT LEADER |
| ‥ | 2025 | TWO DOT LEADER |
| … | 2026 | HORIZONTAL ELLIPSIS |
| ‧ | 2027 | HYPHENATION POINT |
|  | 2028 | LINE SEPARATOR |
|  | 2029 | PARAGRAPH SEPARATOR |
| ‬‬‬ | 202A | LEFT-TO-RIGHT EMBEDDING |
| ‬‬‬ | 202B | RIGHT-TO-LEFT EMBEDDING |
| ‬ | 202C | POP DIRECTIONAL FORMATTING |
| ‬‬‬ | 202D | LEFT-TO-RIGHT OVERRIDE |
| ‬‬‬ | 202E | RIGHT-TO-LEFT OVERRIDE |
|  | 202F | NARROW NON-BREAK SPACE |
| ‰ | 2030 | PER MILLE SIGN |
| ‱ | 2031 | PER TEN THOUSAND SIGN |
| ′ | 2032 | PRIME |
| ″ | 2033 | DOUBLE PRIME |
| ‴ | 2034 | TRIPLE PRIME |
| ‵ | 2035 | REVERSED PRIME |
| ‶ | 2036 | REVERSED DOUBLE PRIME |
| ‷ | 2037 | REVERSED TRIPLE PRIME |
| ‸ | 2038 | CARET |
| ‹ | 2039 | SINGLE LEFT-POINTING ANGLE QUOTATION MARK |
| › | 203A | SINGLE RIGHT-POINTING ANGLE QUOTATION MARK |
| ※ | 203B | REFERENCE MARK |
| ‼ | 203C | DOUBLE EXCLAMATION MARK |
| ‽ | 203D | INTERROBANG |
| ‾ | 203E | OVERLINE |
| ‿ | 203F | UNDERTIE |
| ⁀ | 2040 | CHARACTER TIE |
| ⁁ | 2041 | CARET INSERTION POINT |
| ⁂ | 2042 | ASTERISM |
| ⁃ | 2043 | HYPHEN BULLET |
| ⁄ | 2044 | FRACTION SLASH |
| ⁅ | 2045 | LEFT SQUARE BRACKET WITH QUILL |
| ⁆ | 2046 | RIGHT SQUARE BRACKET WITH QUILL |
| ⁇ | 2047 | DOUBLE QUESTION MARK |
| ⁈ | 2048 | QUESTION EXCLAMATION MARK |
| ⁉ | 2049 | EXCLAMATION QUESTION MARK |
| ⁊ | 204A | TIRONIAN SIGN ET |
| ⁋ | 204B | REVERSED PILCROW SIGN |
| ⁌ | 204C | BLACK LEFTWARDS BULLET |
| ⁍ | 204D | BLACK RIGHTWARDS BULLET |
| ⁎ | 204E | LOW ASTERISK |
| ⁏ | 204F | REVERSED SEMICOLON |
| ⁐ | 2050 | CLOSE UP |
| ⁑ | 2051 | TWO ASTERISKS ALIGNED VERTICALLY |
| ⁒ | 2052 | COMMERCIAL MINUS SIGN |
| ⁓ | 2053 | SWUNG DASH |
| ⁔ | 2054 | INVERTED UNDERTIE |
| ⁕ | 2055 | FLOWER PUNCTUATION MARK |
| ⁖ | 2056 | THREE DOT PUNCTUATION |
| ⁗ | 2057 | QUADRUPLE PRIME |
| ⁘ | 2058 | FOUR DOT PUNCTUATION |
| ⁙ | 2059 | FIVE DOT PUNCTUATION |
| ⁚ | 205A | TWO DOT PUNCTUATION |
| ⁛ | 205B | FOUR DOT MARK |
| ⁜ | 205C | DOTTED CROSS |
| ⁝ | 205D | TRICOLON |
| ⁞ | 205E | VERTICAL FOUR DOTS |
|  | 205F | MEDIUM MATHEMATICAL SPACE |
| ⁠ | 2060 | WORD JOINER |
| ⁡ | 2061 | FUNCTION APPLICATION |
| ⁢ | 2062 | INVISIBLE TIMES |
| ⁣ | 2063 | INVISIBLE SEPARATOR |
| ⁤ | 2064 | INVISIBLE PLUS |
| ⁦ | 2066 | LEFT-TO-RIGHT ISOLATE |
| ⁧ | 2067 | RIGHT-TO-LEFT ISOLATE |
| ⁨ | 2068 | FIRST STRONG ISOLATE |
| ⁩ | 2069 | POP DIRECTIONAL ISOLATE |
| ⁪ | 206A | INHIBIT SYMMETRIC SWAPPING |
| ⁫ | 206B | ACTIVATE SYMMETRIC SWAPPING |
| ⁬ | 206C | INHIBIT ARABIC FORM SHAPING |
| ⁭ | 206D | ACTIVATE ARABIC FORM SHAPING |
| ⁮ | 206E | NATIONAL DIGIT SHAPES |
| ⁯ | 206F | NOMINAL DIGIT SHAPES |
| ₠ | 20A0 | EURO-CURRENCY SIGN |
| ₡ | 20A1 | COLON SIGN |
| ₢ | 20A2 | CRUZEIRO SIGN |
| ₣ | 20A3 | FRENCH FRANC SIGN |
| ₤ | 20A4 | LIRA SIGN |
| ₥ | 20A5 | MILL SIGN |
| ₦ | 20A6 | NAIRA SIGN |
| ₧ | 20A7 | PESETA SIGN |
| ₨ | 20A8 | RUPEE SIGN |
| ₩ | 20A9 | WON SIGN |
| ₪ | 20AA | NEW SHEQEL SIGN |
| ₫ | 20AB | DONG SIGN |
| € | 20AC | EURO SIGN |
| ₭ | 20AD | KIP SIGN |
| ₮ | 20AE | TUGRIK SIGN |
| ₯ | 20AF | DRACHMA SIGN |
| ₰ | 20B0 | GERMAN PENNY SYMBOL |
| ₱ | 20B1 | PESO SIGN |
| ₲ | 20B2 | GUARANI SIGN |
| ₳ | 20B3 | AUSTRAL SIGN |
| ₴ | 20B4 | HRYVNIA SIGN |
| ₵ | 20B5 | CEDI SIGN |
| ₶ | 20B6 | LIVRE TOURNOIS SIGN |
| ₷ | 20B7 | SPESMILO SIGN |
| ₸ | 20B8 | TENGE SIGN |
| ₹ | 20B9 | INDIAN RUPEE SIGN |
| ℀ | 2100 | ACCOUNT OF |
| ℁ | 2101 | ADDRESSED TO THE SUBJECT |
| ℂ | 2102 | DOUBLE-STRUCK CAPITAL C |
| ℃ | 2103 | DEGREE CELSIUS |
| ℄ | 2104 | CENTRE LINE SYMBOL |
| ℅ | 2105 | CARE OF |
| ℆ | 2106 | CADA UNA |
| ℇ | 2107 | EULER CONSTANT |
| ℈ | 2108 | SCRUPLE |
| ℉ | 2109 | DEGREE FAHRENHEIT |
| ℊ | 210A | SCRIPT SMALL G |
| ℋ | 210B | SCRIPT CAPITAL H |
| ℌ | 210C | BLACK-LETTER CAPITAL H |
| ℍ | 210D | DOUBLE-STRUCK CAPITAL H |
| ℎ | 210E | PLANCK CONSTANT |
| ℏ | 210F | PLANCK CONSTANT OVER TWO PI |
| ℐ | 2110 | SCRIPT CAPITAL I |
| ℑ | 2111 | BLACK-LETTER CAPITAL I |
| ℒ | 2112 | SCRIPT CAPITAL L |
| ℓ | 2113 | SCRIPT SMALL L |
| ℔ | 2114 | L B BAR SYMBOL |
| ℕ | 2115 | DOUBLE-STRUCK CAPITAL N |
| № | 2116 | NUMERO SIGN |
| ℗ | 2117 | SOUND RECORDING COPYRIGHT |
| ℘ | 2118 | SCRIPT CAPITAL P |
| ℙ | 2119 | DOUBLE-STRUCK CAPITAL P |
| ℚ | 211A | DOUBLE-STRUCK CAPITAL Q |
| ℛ | 211B | SCRIPT CAPITAL R |
| ℜ | 211C | BLACK-LETTER CAPITAL R |
| ℝ | 211D | DOUBLE-STRUCK CAPITAL R |
| ℞ | 211E | PRESCRIPTION TAKE |
| ℟ | 211F | RESPONSE |
| ℠ | 2120 | SERVICE MARK |
| ℡ | 2121 | TELEPHONE SIGN |
| ™ | 2122 | TRADE MARK SIGN |
| ℣ | 2123 | VERSICLE |
| ℤ | 2124 | DOUBLE-STRUCK CAPITAL Z |
| ℥ | 2125 | OUNCE SIGN |
| Ω | 2126 | OHM SIGN |
| ℧ | 2127 | INVERTED OHM SIGN |
| ℨ | 2128 | BLACK-LETTER CAPITAL Z |
| ℩ | 2129 | TURNED GREEK SMALL LETTER IOTA |
| K | 212A | KELVIN SIGN |
| Å | 212B | ANGSTROM SIGN |
| ℬ | 212C | SCRIPT CAPITAL B |
| ℭ | 212D | BLACK-LETTER CAPITAL C |
| ℮ | 212E | ESTIMATED SYMBOL |
| ℯ | 212F | SCRIPT SMALL E |
| ℰ | 2130 | SCRIPT CAPITAL E |
| ℱ | 2131 | SCRIPT CAPITAL F |
| Ⅎ | 2132 | TURNED CAPITAL F |
| ℳ | 2133 | SCRIPT CAPITAL M |
| ℴ | 2134 | SCRIPT SMALL O |
| ℵ | 2135 | ALEF SYMBOL |
| ℶ | 2136 | BET SYMBOL |
| ℷ | 2137 | GIMEL SYMBOL |
| ℸ | 2138 | DALET SYMBOL |
| ℹ | 2139 | INFORMATION SOURCE |
| ℺ | 213A | ROTATED CAPITAL Q |
| ℻ | 213B | FACSIMILE SIGN |
| ℼ | 213C | DOUBLE-STRUCK SMALL PI |
| ℽ | 213D | DOUBLE-STRUCK SMALL GAMMA |
| ℾ | 213E | DOUBLE-STRUCK CAPITAL GAMMA |
| ℿ | 213F | DOUBLE-STRUCK CAPITAL PI |
| ⅀ | 2140 | DOUBLE-STRUCK N-ARY SUMMATION |
| ⅁ | 2141 | TURNED SANS-SERIF CAPITAL G |
| ⅂ | 2142 | TURNED SANS-SERIF CAPITAL L |
| ⅃ | 2143 | REVERSED SANS-SERIF CAPITAL L |
| ⅄ | 2144 | TURNED SANS-SERIF CAPITAL Y |
| ⅅ | 2145 | DOUBLE-STRUCK ITALIC CAPITAL D |
| ⅆ | 2146 | DOUBLE-STRUCK ITALIC SMALL D |
| ⅇ | 2147 | DOUBLE-STRUCK ITALIC SMALL E |
| ⅈ | 2148 | DOUBLE-STRUCK ITALIC SMALL I |
| ⅉ | 2149 | DOUBLE-STRUCK ITALIC SMALL J |
| ⅊ | 214A | PROPERTY LINE |
| ⅋ | 214B | TURNED AMPERSAND |
| ⅌ | 214C | PER SIGN |
| ⅍ | 214D | AKTIESELSKAB |
| ⅎ | 214E | TURNED SMALL F |
| ⅏ | 214F | SYMBOL FOR SAMARITAN SOURCE |
| ← | 2190 | LEFTWARDS ARROW |
| ↑ | 2191 | UPWARDS ARROW |
| → | 2192 | RIGHTWARDS ARROW |
| ↓ | 2193 | DOWNWARDS ARROW |
| ↔ | 2194 | LEFT RIGHT ARROW |
| ↕ | 2195 | UP DOWN ARROW |
| ↖ | 2196 | NORTH WEST ARROW |
| ↗ | 2197 | NORTH EAST ARROW |
| ↘ | 2198 | SOUTH EAST ARROW |
| ↙ | 2199 | SOUTH WEST ARROW |
| ↚ | 219A | LEFTWARDS ARROW WITH STROKE |
| ↛ | 219B | RIGHTWARDS ARROW WITH STROKE |
| ↜ | 219C | LEFTWARDS WAVE ARROW |
| ↝ | 219D | RIGHTWARDS WAVE ARROW |
| ↞ | 219E | LEFTWARDS TWO HEADED ARROW |
| ↟ | 219F | UPWARDS TWO HEADED ARROW |
| ↠ | 21A0 | RIGHTWARDS TWO HEADED ARROW |
| ↡ | 21A1 | DOWNWARDS TWO HEADED ARROW |
| ↢ | 21A2 | LEFTWARDS ARROW WITH TAIL |
| ↣ | 21A3 | RIGHTWARDS ARROW WITH TAIL |
| ↤ | 21A4 | LEFTWARDS ARROW FROM BAR |
| ↥ | 21A5 | UPWARDS ARROW FROM BAR |
| ↦ | 21A6 | RIGHTWARDS ARROW FROM BAR |
| ↧ | 21A7 | DOWNWARDS ARROW FROM BAR |
| ↨ | 21A8 | UP DOWN ARROW WITH BASE |
| ↩ | 21A9 | LEFTWARDS ARROW WITH HOOK |
| ↪ | 21AA | RIGHTWARDS ARROW WITH HOOK |
| ↫ | 21AB | LEFTWARDS ARROW WITH LOOP |
| ↬ | 21AC | RIGHTWARDS ARROW WITH LOOP |
| ↭ | 21AD | LEFT RIGHT WAVE ARROW |
| ↮ | 21AE | LEFT RIGHT ARROW WITH STROKE |
| ↯ | 21AF | DOWNWARDS ZIGZAG ARROW |
| ↰ | 21B0 | UPWARDS ARROW WITH TIP LEFTWARDS |
| ↱ | 21B1 | UPWARDS ARROW WITH TIP RIGHTWARDS |
| ↲ | 21B2 | DOWNWARDS ARROW WITH TIP LEFTWARDS |
| ↳ | 21B3 | DOWNWARDS ARROW WITH TIP RIGHTWARDS |
| ↴ | 21B4 | RIGHTWARDS ARROW WITH CORNER DOWNWARDS |
| ↵ | 21B5 | DOWNWARDS ARROW WITH CORNER LEFTWARDS |
| ↶ | 21B6 | ANTICLOCKWISE TOP SEMICIRCLE ARROW |
| ↷ | 21B7 | CLOCKWISE TOP SEMICIRCLE ARROW |
| ↸ | 21B8 | NORTH WEST ARROW TO LONG BAR |
| ↹ | 21B9 | LEFTWARDS ARROW TO BAR OVER RIGHTWARDS ARROW TO BAR |
| ↺ | 21BA | ANTICLOCKWISE OPEN CIRCLE ARROW |
| ↻ | 21BB | CLOCKWISE OPEN CIRCLE ARROW |
| ↼ | 21BC | LEFTWARDS HARPOON WITH BARB UPWARDS |
| ↽ | 21BD | LEFTWARDS HARPOON WITH BARB DOWNWARDS |
| ↾ | 21BE | UPWARDS HARPOON WITH BARB RIGHTWARDS |
| ↿ | 21BF | UPWARDS HARPOON WITH BARB LEFTWARDS |
| ⇀ | 21C0 | RIGHTWARDS HARPOON WITH BARB UPWARDS |
| ⇁ | 21C1 | RIGHTWARDS HARPOON WITH BARB DOWNWARDS |
| ⇂ | 21C2 | DOWNWARDS HARPOON WITH BARB RIGHTWARDS |
| ⇃ | 21C3 | DOWNWARDS HARPOON WITH BARB LEFTWARDS |
| ⇄ | 21C4 | RIGHTWARDS ARROW OVER LEFTWARDS ARROW |
| ⇅ | 21C5 | UPWARDS ARROW LEFTWARDS OF DOWNWARDS ARROW |
| ⇆ | 21C6 | LEFTWARDS ARROW OVER RIGHTWARDS ARROW |
| ⇇ | 21C7 | LEFTWARDS PAIRED ARROWS |
| ⇈ | 21C8 | UPWARDS PAIRED ARROWS |
| ⇉ | 21C9 | RIGHTWARDS PAIRED ARROWS |
| ⇊ | 21CA | DOWNWARDS PAIRED ARROWS |
| ⇋ | 21CB | LEFTWARDS HARPOON OVER RIGHTWARDS HARPOON |
| ⇌ | 21CC | RIGHTWARDS HARPOON OVER LEFTWARDS HARPOON |
| ⇍ | 21CD | LEFTWARDS DOUBLE ARROW WITH STROKE |
| ⇎ | 21CE | LEFT RIGHT DOUBLE ARROW WITH STROKE |
| ⇏ | 21CF | RIGHTWARDS DOUBLE ARROW WITH STROKE |
| ⇐ | 21D0 | LEFTWARDS DOUBLE ARROW |
| ⇑ | 21D1 | UPWARDS DOUBLE ARROW |
| ⇒ | 21D2 | RIGHTWARDS DOUBLE ARROW |
| ⇓ | 21D3 | DOWNWARDS DOUBLE ARROW |
| ⇔ | 21D4 | LEFT RIGHT DOUBLE ARROW |
| ⇕ | 21D5 | UP DOWN DOUBLE ARROW |
| ⇖ | 21D6 | NORTH WEST DOUBLE ARROW |
| ⇗ | 21D7 | NORTH EAST DOUBLE ARROW |
| ⇘ | 21D8 | SOUTH EAST DOUBLE ARROW |
| ⇙ | 21D9 | SOUTH WEST DOUBLE ARROW |
| ⇚ | 21DA | LEFTWARDS TRIPLE ARROW |
| ⇛ | 21DB | RIGHTWARDS TRIPLE ARROW |
| ⇜ | 21DC | LEFTWARDS SQUIGGLE ARROW |
| ⇝ | 21DD | RIGHTWARDS SQUIGGLE ARROW |
| ⇞ | 21DE | UPWARDS ARROW WITH DOUBLE STROKE |
| ⇟ | 21DF | DOWNWARDS ARROW WITH DOUBLE STROKE |
| ⇠ | 21E0 | LEFTWARDS DASHED ARROW |
| ⇡ | 21E1 | UPWARDS DASHED ARROW |
| ⇢ | 21E2 | RIGHTWARDS DASHED ARROW |
| ⇣ | 21E3 | DOWNWARDS DASHED ARROW |
| ⇤ | 21E4 | LEFTWARDS ARROW TO BAR |
| ⇥ | 21E5 | RIGHTWARDS ARROW TO BAR |
| ⇦ | 21E6 | LEFTWARDS WHITE ARROW |
| ⇧ | 21E7 | UPWARDS WHITE ARROW |
| ⇨ | 21E8 | RIGHTWARDS WHITE ARROW |
| ⇩ | 21E9 | DOWNWARDS WHITE ARROW |
| ⇪ | 21EA | UPWARDS WHITE ARROW FROM BAR |
| ⇫ | 21EB | UPWARDS WHITE ARROW ON PEDESTAL |
| ⇬ | 21EC | UPWARDS WHITE ARROW ON PEDESTAL WITH HORIZONTAL BAR |
| ⇭ | 21ED | UPWARDS WHITE ARROW ON PEDESTAL WITH VERTICAL BAR |
| ⇮ | 21EE | UPWARDS WHITE DOUBLE ARROW |
| ⇯ | 21EF | UPWARDS WHITE DOUBLE ARROW ON PEDESTAL |
| ⇰ | 21F0 | RIGHTWARDS WHITE ARROW FROM WALL |
| ⇱ | 21F1 | NORTH WEST ARROW TO CORNER |
| ⇲ | 21F2 | SOUTH EAST ARROW TO CORNER |
| ⇳ | 21F3 | UP DOWN WHITE ARROW |
| ⇴ | 21F4 | RIGHT ARROW WITH SMALL CIRCLE |
| ⇵ | 21F5 | DOWNWARDS ARROW LEFTWARDS OF UPWARDS ARROW |
| ⇶ | 21F6 | THREE RIGHTWARDS ARROWS |
| ⇷ | 21F7 | LEFTWARDS ARROW WITH VERTICAL STROKE |
| ⇸ | 21F8 | RIGHTWARDS ARROW WITH VERTICAL STROKE |
| ⇹ | 21F9 | LEFT RIGHT ARROW WITH VERTICAL STROKE |
| ⇺ | 21FA | LEFTWARDS ARROW WITH DOUBLE VERTICAL STROKE |
| ⇻ | 21FB | RIGHTWARDS ARROW WITH DOUBLE VERTICAL STROKE |
| ⇼ | 21FC | LEFT RIGHT ARROW WITH DOUBLE VERTICAL STROKE |
| ⇽ | 21FD | LEFTWARDS OPEN-HEADED ARROW |
| ⇾ | 21FE | RIGHTWARDS OPEN-HEADED ARROW |
| ⇿ | 21FF | LEFT RIGHT OPEN-HEADED ARROW |
| ∀ | 2200 | FOR ALL |
| ∁ | 2201 | COMPLEMENT |
| ∂ | 2202 | PARTIAL DIFFERENTIAL |
| ∃ | 2203 | THERE EXISTS |
| ∄ | 2204 | THERE DOES NOT EXIST |
| ∅ | 2205 | EMPTY SET |
| ∆ | 2206 | INCREMENT |
| ∇ | 2207 | NABLA |
| ∈ | 2208 | ELEMENT OF |
| ∉ | 2209 | NOT AN ELEMENT OF |
| ∊ | 220A | SMALL ELEMENT OF |
| ∋ | 220B | CONTAINS AS MEMBER |
| ∌ | 220C | DOES NOT CONTAIN AS MEMBER |
| ∍ | 220D | SMALL CONTAINS AS MEMBER |
| ∎ | 220E | END OF PROOF |
| ∏ | 220F | N-ARY PRODUCT |
| ∐ | 2210 | N-ARY COPRODUCT |
| ∑ | 2211 | N-ARY SUMMATION |
| − | 2212 | MINUS SIGN |
| ∓ | 2213 | MINUS-OR-PLUS SIGN |
| ∔ | 2214 | DOT PLUS |
| ∕ | 2215 | DIVISION SLASH |
| ∖ | 2216 | SET MINUS |
| ∗ | 2217 | ASTERISK OPERATOR |
| ∘ | 2218 | RING OPERATOR |
| ∙ | 2219 | BULLET OPERATOR |
| √ | 221A | SQUARE ROOT |
| ∛ | 221B | CUBE ROOT |
| ∜ | 221C | FOURTH ROOT |
| ∝ | 221D | PROPORTIONAL TO |
| ∞ | 221E | INFINITY |
| ∟ | 221F | RIGHT ANGLE |
| ∠ | 2220 | ANGLE |
| ∡ | 2221 | MEASURED ANGLE |
| ∢ | 2222 | SPHERICAL ANGLE |
| ∣ | 2223 | DIVIDES |
| ∤ | 2224 | DOES NOT DIVIDE |
| ∥ | 2225 | PARALLEL TO |
| ∦ | 2226 | NOT PARALLEL TO |
| ∧ | 2227 | LOGICAL AND |
| ∨ | 2228 | LOGICAL OR |
| ∩ | 2229 | INTERSECTION |
| ∪ | 222A | UNION |
| ∫ | 222B | INTEGRAL |
| ∬ | 222C | DOUBLE INTEGRAL |
| ∭ | 222D | TRIPLE INTEGRAL |
| ∮ | 222E | CONTOUR INTEGRAL |
| ∯ | 222F | SURFACE INTEGRAL |
| ∰ | 2230 | VOLUME INTEGRAL |
| ∱ | 2231 | CLOCKWISE INTEGRAL |
| ∲ | 2232 | CLOCKWISE CONTOUR INTEGRAL |
| ∳ | 2233 | ANTICLOCKWISE CONTOUR INTEGRAL |
| ∴ | 2234 | THEREFORE |
| ∵ | 2235 | BECAUSE |
| ∶ | 2236 | RATIO |
| ∷ | 2237 | PROPORTION |
| ∸ | 2238 | DOT MINUS |
| ∹ | 2239 | EXCESS |
| ∺ | 223A | GEOMETRIC PROPORTION |
| ∻ | 223B | HOMOTHETIC |
| ∼ | 223C | TILDE OPERATOR |
| ∽ | 223D | REVERSED TILDE |
| ∾ | 223E | INVERTED LAZY S |
| ∿ | 223F | SINE WAVE |
| ≀ | 2240 | WREATH PRODUCT |
| ≁ | 2241 | NOT TILDE |
| ≂ | 2242 | MINUS TILDE |
| ≃ | 2243 | ASYMPTOTICALLY EQUAL TO |
| ≄ | 2244 | NOT ASYMPTOTICALLY EQUAL TO |
| ≅ | 2245 | APPROXIMATELY EQUAL TO |
| ≆ | 2246 | APPROXIMATELY BUT NOT ACTUALLY EQUAL TO |
| ≇ | 2247 | NEITHER APPROXIMATELY NOR ACTUALLY EQUAL TO |
| ≈ | 2248 | ALMOST EQUAL TO |
| ≉ | 2249 | NOT ALMOST EQUAL TO |
| ≊ | 224A | ALMOST EQUAL OR EQUAL TO |
| ≋ | 224B | TRIPLE TILDE |
| ≌ | 224C | ALL EQUAL TO |
| ≍ | 224D | EQUIVALENT TO |
| ≎ | 224E | GEOMETRICALLY EQUIVALENT TO |
| ≏ | 224F | DIFFERENCE BETWEEN |
| ≐ | 2250 | APPROACHES THE LIMIT |
| ≑ | 2251 | GEOMETRICALLY EQUAL TO |
| ≒ | 2252 | APPROXIMATELY EQUAL TO OR THE IMAGE OF |
| ≓ | 2253 | IMAGE OF OR APPROXIMATELY EQUAL TO |
| ≔ | 2254 | COLON EQUALS |
| ≕ | 2255 | EQUALS COLON |
| ≖ | 2256 | RING IN EQUAL TO |
| ≗ | 2257 | RING EQUAL TO |
| ≘ | 2258 | CORRESPONDS TO |
| ≙ | 2259 | ESTIMATES |
| ≚ | 225A | EQUIANGULAR TO |
| ≛ | 225B | STAR EQUALS |
| ≜ | 225C | DELTA EQUAL TO |
| ≝ | 225D | EQUAL TO BY DEFINITION |
| ≞ | 225E | MEASURED BY |
| ≟ | 225F | QUESTIONED EQUAL TO |
| ≠ | 2260 | NOT EQUAL TO |
| ≡ | 2261 | IDENTICAL TO |
| ≢ | 2262 | NOT IDENTICAL TO |
| ≣ | 2263 | STRICTLY EQUIVALENT TO |
| ≤ | 2264 | LESS-THAN OR EQUAL TO |
| ≥ | 2265 | GREATER-THAN OR EQUAL TO |
| ≦ | 2266 | LESS-THAN OVER EQUAL TO |
| ≧ | 2267 | GREATER-THAN OVER EQUAL TO |
| ≨ | 2268 | LESS-THAN BUT NOT EQUAL TO |
| ≩ | 2269 | GREATER-THAN BUT NOT EQUAL TO |
| ≪ | 226A | MUCH LESS-THAN |
| ≫ | 226B | MUCH GREATER-THAN |
| ≬ | 226C | BETWEEN |
| ≭ | 226D | NOT EQUIVALENT TO |
| ≮ | 226E | NOT LESS-THAN |
| ≯ | 226F | NOT GREATER-THAN |
| ≰ | 2270 | NEITHER LESS-THAN NOR EQUAL TO |
| ≱ | 2271 | NEITHER GREATER-THAN NOR EQUAL TO |
| ≲ | 2272 | LESS-THAN OR EQUIVALENT TO |
| ≳ | 2273 | GREATER-THAN OR EQUIVALENT TO |
| ≴ | 2274 | NEITHER LESS-THAN NOR EQUIVALENT TO |
| ≵ | 2275 | NEITHER GREATER-THAN NOR EQUIVALENT TO |
| ≶ | 2276 | LESS-THAN OR GREATER-THAN |
| ≷ | 2277 | GREATER-THAN OR LESS-THAN |
| ≸ | 2278 | NEITHER LESS-THAN NOR GREATER-THAN |
| ≹ | 2279 | NEITHER GREATER-THAN NOR LESS-THAN |
| ≺ | 227A | PRECEDES |
| ≻ | 227B | SUCCEEDS |
| ≼ | 227C | PRECEDES OR EQUAL TO |
| ≽ | 227D | SUCCEEDS OR EQUAL TO |
| ≾ | 227E | PRECEDES OR EQUIVALENT TO |
| ≿ | 227F | SUCCEEDS OR EQUIVALENT TO |
| ⊀ | 2280 | DOES NOT PRECEDE |
| ⊁ | 2281 | DOES NOT SUCCEED |
| ⊂ | 2282 | SUBSET OF |
| ⊃ | 2283 | SUPERSET OF |
| ⊄ | 2284 | NOT A SUBSET OF |
| ⊅ | 2285 | NOT A SUPERSET OF |
| ⊆ | 2286 | SUBSET OF OR EQUAL TO |
| ⊇ | 2287 | SUPERSET OF OR EQUAL TO |
| ⊈ | 2288 | NEITHER A SUBSET OF NOR EQUAL TO |
| ⊉ | 2289 | NEITHER A SUPERSET OF NOR EQUAL TO |
| ⊊ | 228A | SUBSET OF WITH NOT EQUAL TO |
| ⊋ | 228B | SUPERSET OF WITH NOT EQUAL TO |
| ⊌ | 228C | MULTISET |
| ⊍ | 228D | MULTISET MULTIPLICATION |
| ⊎ | 228E | MULTISET UNION |
| ⊏ | 228F | SQUARE IMAGE OF |
| ⊐ | 2290 | SQUARE ORIGINAL OF |
| ⊑ | 2291 | SQUARE IMAGE OF OR EQUAL TO |
| ⊒ | 2292 | SQUARE ORIGINAL OF OR EQUAL TO |
| ⊓ | 2293 | SQUARE CAP |
| ⊔ | 2294 | SQUARE CUP |
| ⊕ | 2295 | CIRCLED PLUS |
| ⊖ | 2296 | CIRCLED MINUS |
| ⊗ | 2297 | CIRCLED TIMES |
| ⊘ | 2298 | CIRCLED DIVISION SLASH |
| ⊙ | 2299 | CIRCLED DOT OPERATOR |
| ⊚ | 229A | CIRCLED RING OPERATOR |
| ⊛ | 229B | CIRCLED ASTERISK OPERATOR |
| ⊜ | 229C | CIRCLED EQUALS |
| ⊝ | 229D | CIRCLED DASH |
| ⊞ | 229E | SQUARED PLUS |
| ⊟ | 229F | SQUARED MINUS |
| ⊠ | 22A0 | SQUARED TIMES |
| ⊡ | 22A1 | SQUARED DOT OPERATOR |
| ⊢ | 22A2 | RIGHT TACK |
| ⊣ | 22A3 | LEFT TACK |
| ⊤ | 22A4 | DOWN TACK |
| ⊥ | 22A5 | UP TACK |
| ⊦ | 22A6 | ASSERTION |
| ⊧ | 22A7 | MODELS |
| ⊨ | 22A8 | TRUE |
| ⊩ | 22A9 | FORCES |
| ⊪ | 22AA | TRIPLE VERTICAL BAR RIGHT TURNSTILE |
| ⊫ | 22AB | DOUBLE VERTICAL BAR DOUBLE RIGHT TURNSTILE |
| ⊬ | 22AC | DOES NOT PROVE |
| ⊭ | 22AD | NOT TRUE |
| ⊮ | 22AE | DOES NOT FORCE |
| ⊯ | 22AF | NEGATED DOUBLE VERTICAL BAR DOUBLE RIGHT TURNSTILE |
| ⊰ | 22B0 | PRECEDES UNDER RELATION |
| ⊱ | 22B1 | SUCCEEDS UNDER RELATION |
| ⊲ | 22B2 | NORMAL SUBGROUP OF |
| ⊳ | 22B3 | CONTAINS AS NORMAL SUBGROUP |
| ⊴ | 22B4 | NORMAL SUBGROUP OF OR EQUAL TO |
| ⊵ | 22B5 | CONTAINS AS NORMAL SUBGROUP OR EQUAL TO |
| ⊶ | 22B6 | ORIGINAL OF |
| ⊷ | 22B7 | IMAGE OF |
| ⊸ | 22B8 | MULTIMAP |
| ⊹ | 22B9 | HERMITIAN CONJUGATE MATRIX |
| ⊺ | 22BA | INTERCALATE |
| ⊻ | 22BB | XOR |
| ⊼ | 22BC | NAND |
| ⊽ | 22BD | NOR |
| ⊾ | 22BE | RIGHT ANGLE WITH ARC |
| ⊿ | 22BF | RIGHT TRIANGLE |
| ⋀ | 22C0 | N-ARY LOGICAL AND |
| ⋁ | 22C1 | N-ARY LOGICAL OR |
| ⋂ | 22C2 | N-ARY INTERSECTION |
| ⋃ | 22C3 | N-ARY UNION |
| ⋄ | 22C4 | DIAMOND OPERATOR |
| ⋅ | 22C5 | DOT OPERATOR |
| ⋆ | 22C6 | STAR OPERATOR |
| ⋇ | 22C7 | DIVISION TIMES |
| ⋈ | 22C8 | BOWTIE |
| ⋉ | 22C9 | LEFT NORMAL FACTOR SEMIDIRECT PRODUCT |
| ⋊ | 22CA | RIGHT NORMAL FACTOR SEMIDIRECT PRODUCT |
| ⋋ | 22CB | LEFT SEMIDIRECT PRODUCT |
| ⋌ | 22CC | RIGHT SEMIDIRECT PRODUCT |
| ⋍ | 22CD | REVERSED TILDE EQUALS |
| ⋎ | 22CE | CURLY LOGICAL OR |
| ⋏ | 22CF | CURLY LOGICAL AND |
| ⋐ | 22D0 | DOUBLE SUBSET |
| ⋑ | 22D1 | DOUBLE SUPERSET |
| ⋒ | 22D2 | DOUBLE INTERSECTION |
| ⋓ | 22D3 | DOUBLE UNION |
| ⋔ | 22D4 | PITCHFORK |
| ⋕ | 22D5 | EQUAL AND PARALLEL TO |
| ⋖ | 22D6 | LESS-THAN WITH DOT |
| ⋗ | 22D7 | GREATER-THAN WITH DOT |
| ⋘ | 22D8 | VERY MUCH LESS-THAN |
| ⋙ | 22D9 | VERY MUCH GREATER-THAN |
| ⋚ | 22DA | LESS-THAN EQUAL TO OR GREATER-THAN |
| ⋛ | 22DB | GREATER-THAN EQUAL TO OR LESS-THAN |
| ⋜ | 22DC | EQUAL TO OR LESS-THAN |
| ⋝ | 22DD | EQUAL TO OR GREATER-THAN |
| ⋞ | 22DE | EQUAL TO OR PRECEDES |
| ⋟ | 22DF | EQUAL TO OR SUCCEEDS |
| ⋠ | 22E0 | DOES NOT PRECEDE OR EQUAL |
| ⋡ | 22E1 | DOES NOT SUCCEED OR EQUAL |
| ⋢ | 22E2 | NOT SQUARE IMAGE OF OR EQUAL TO |
| ⋣ | 22E3 | NOT SQUARE ORIGINAL OF OR EQUAL TO |
| ⋤ | 22E4 | SQUARE IMAGE OF OR NOT EQUAL TO |
| ⋥ | 22E5 | SQUARE ORIGINAL OF OR NOT EQUAL TO |
| ⋦ | 22E6 | LESS-THAN BUT NOT EQUIVALENT TO |
| ⋧ | 22E7 | GREATER-THAN BUT NOT EQUIVALENT TO |
| ⋨ | 22E8 | PRECEDES BUT NOT EQUIVALENT TO |
| ⋩ | 22E9 | SUCCEEDS BUT NOT EQUIVALENT TO |
| ⋪ | 22EA | NOT NORMAL SUBGROUP OF |
| ⋫ | 22EB | DOES NOT CONTAIN AS NORMAL SUBGROUP |
| ⋬ | 22EC | NOT NORMAL SUBGROUP OF OR EQUAL TO |
| ⋭ | 22ED | DOES NOT CONTAIN AS NORMAL SUBGROUP OR EQUAL |
| ⋮ | 22EE | VERTICAL ELLIPSIS |
| ⋯ | 22EF | MIDLINE HORIZONTAL ELLIPSIS |
| ⋰ | 22F0 | UP RIGHT DIAGONAL ELLIPSIS |
| ⋱ | 22F1 | DOWN RIGHT DIAGONAL ELLIPSIS |
| ⋲ | 22F2 | ELEMENT OF WITH LONG HORIZONTAL STROKE |
| ⋳ | 22F3 | ELEMENT OF WITH VERTICAL BAR AT END OF HORIZONTAL STROKE |
| ⋴ | 22F4 | SMALL ELEMENT OF WITH VERTICAL BAR AT END OF HORIZONTAL STROKE |
| ⋵ | 22F5 | ELEMENT OF WITH DOT ABOVE |
| ⋶ | 22F6 | ELEMENT OF WITH OVERBAR |
| ⋷ | 22F7 | SMALL ELEMENT OF WITH OVERBAR |
| ⋸ | 22F8 | ELEMENT OF WITH UNDERBAR |
| ⋹ | 22F9 | ELEMENT OF WITH TWO HORIZONTAL STROKES |
| ⋺ | 22FA | CONTAINS WITH LONG HORIZONTAL STROKE |
| ⋻ | 22FB | CONTAINS WITH VERTICAL BAR AT END OF HORIZONTAL STROKE |
| ⋼ | 22FC | SMALL CONTAINS WITH VERTICAL BAR AT END OF HORIZONTAL STROKE |
| ⋽ | 22FD | CONTAINS WITH OVERBAR |
| ⋾ | 22FE | SMALL CONTAINS WITH OVERBAR |
| ⋿ | 22FF | Z NOTATION BAG MEMBERSHIP |
| ─ | 2500 | BOX DRAWINGS LIGHT HORIZONTAL |
| ━ | 2501 | BOX DRAWINGS HEAVY HORIZONTAL |
| │ | 2502 | BOX DRAWINGS LIGHT VERTICAL |
| ┃ | 2503 | BOX DRAWINGS HEAVY VERTICAL |
| ┄ | 2504 | BOX DRAWINGS LIGHT TRIPLE DASH HORIZONTAL |
| ┅ | 2505 | BOX DRAWINGS HEAVY TRIPLE DASH HORIZONTAL |
| ┆ | 2506 | BOX DRAWINGS LIGHT TRIPLE DASH VERTICAL |
| ┇ | 2507 | BOX DRAWINGS HEAVY TRIPLE DASH VERTICAL |
| ┈ | 2508 | BOX DRAWINGS LIGHT QUADRUPLE DASH HORIZONTAL |
| ┉ | 2509 | BOX DRAWINGS HEAVY QUADRUPLE DASH HORIZONTAL |
| ┊ | 250A | BOX DRAWINGS LIGHT QUADRUPLE DASH VERTICAL |
| ┋ | 250B | BOX DRAWINGS HEAVY QUADRUPLE DASH VERTICAL |
| ┌ | 250C | BOX DRAWINGS LIGHT DOWN AND RIGHT |
| ┍ | 250D | BOX DRAWINGS DOWN LIGHT AND RIGHT HEAVY |
| ┎ | 250E | BOX DRAWINGS DOWN HEAVY AND RIGHT LIGHT |
| ┏ | 250F | BOX DRAWINGS HEAVY DOWN AND RIGHT |
| ┐ | 2510 | BOX DRAWINGS LIGHT DOWN AND LEFT |
| ┑ | 2511 | BOX DRAWINGS DOWN LIGHT AND LEFT HEAVY |
| ┒ | 2512 | BOX DRAWINGS DOWN HEAVY AND LEFT LIGHT |
| ┓ | 2513 | BOX DRAWINGS HEAVY DOWN AND LEFT |
| └ | 2514 | BOX DRAWINGS LIGHT UP AND RIGHT |
| ┕ | 2515 | BOX DRAWINGS UP LIGHT AND RIGHT HEAVY |
| ┖ | 2516 | BOX DRAWINGS UP HEAVY AND RIGHT LIGHT |
| ┗ | 2517 | BOX DRAWINGS HEAVY UP AND RIGHT |
| ┘ | 2518 | BOX DRAWINGS LIGHT UP AND LEFT |
| ┙ | 2519 | BOX DRAWINGS UP LIGHT AND LEFT HEAVY |
| ┚ | 251A | BOX DRAWINGS UP HEAVY AND LEFT LIGHT |
| ┛ | 251B | BOX DRAWINGS HEAVY UP AND LEFT |
| ├ | 251C | BOX DRAWINGS LIGHT VERTICAL AND RIGHT |
| ┝ | 251D | BOX DRAWINGS VERTICAL LIGHT AND RIGHT HEAVY |
| ┞ | 251E | BOX DRAWINGS UP HEAVY AND RIGHT DOWN LIGHT |
| ┟ | 251F | BOX DRAWINGS DOWN HEAVY AND RIGHT UP LIGHT |
| ┠ | 2520 | BOX DRAWINGS VERTICAL HEAVY AND RIGHT LIGHT |
| ┡ | 2521 | BOX DRAWINGS DOWN LIGHT AND RIGHT UP HEAVY |
| ┢ | 2522 | BOX DRAWINGS UP LIGHT AND RIGHT DOWN HEAVY |
| ┣ | 2523 | BOX DRAWINGS HEAVY VERTICAL AND RIGHT |
| ┤ | 2524 | BOX DRAWINGS LIGHT VERTICAL AND LEFT |
| ┥ | 2525 | BOX DRAWINGS VERTICAL LIGHT AND LEFT HEAVY |
| ┦ | 2526 | BOX DRAWINGS UP HEAVY AND LEFT DOWN LIGHT |
| ┧ | 2527 | BOX DRAWINGS DOWN HEAVY AND LEFT UP LIGHT |
| ┨ | 2528 | BOX DRAWINGS VERTICAL HEAVY AND LEFT LIGHT |
| ┩ | 2529 | BOX DRAWINGS DOWN LIGHT AND LEFT UP HEAVY |
| ┪ | 252A | BOX DRAWINGS UP LIGHT AND LEFT DOWN HEAVY |
| ┫ | 252B | BOX DRAWINGS HEAVY VERTICAL AND LEFT |
| ┬ | 252C | BOX DRAWINGS LIGHT DOWN AND HORIZONTAL |
| ┭ | 252D | BOX DRAWINGS LEFT HEAVY AND RIGHT DOWN LIGHT |
| ┮ | 252E | BOX DRAWINGS RIGHT HEAVY AND LEFT DOWN LIGHT |
| ┯ | 252F | BOX DRAWINGS DOWN LIGHT AND HORIZONTAL HEAVY |
| ┰ | 2530 | BOX DRAWINGS DOWN HEAVY AND HORIZONTAL LIGHT |
| ┱ | 2531 | BOX DRAWINGS RIGHT LIGHT AND LEFT DOWN HEAVY |
| ┲ | 2532 | BOX DRAWINGS LEFT LIGHT AND RIGHT DOWN HEAVY |
| ┳ | 2533 | BOX DRAWINGS HEAVY DOWN AND HORIZONTAL |
| ┴ | 2534 | BOX DRAWINGS LIGHT UP AND HORIZONTAL |
| ┵ | 2535 | BOX DRAWINGS LEFT HEAVY AND RIGHT UP LIGHT |
| ┶ | 2536 | BOX DRAWINGS RIGHT HEAVY AND LEFT UP LIGHT |
| ┷ | 2537 | BOX DRAWINGS UP LIGHT AND HORIZONTAL HEAVY |
| ┸ | 2538 | BOX DRAWINGS UP HEAVY AND HORIZONTAL LIGHT |
| ┹ | 2539 | BOX DRAWINGS RIGHT LIGHT AND LEFT UP HEAVY |
| ┺ | 253A | BOX DRAWINGS LEFT LIGHT AND RIGHT UP HEAVY |
| ┻ | 253B | BOX DRAWINGS HEAVY UP AND HORIZONTAL |
| ┼ | 253C | BOX DRAWINGS LIGHT VERTICAL AND HORIZONTAL |
| ┽ | 253D | BOX DRAWINGS LEFT HEAVY AND RIGHT VERTICAL LIGHT |
| ┾ | 253E | BOX DRAWINGS RIGHT HEAVY AND LEFT VERTICAL LIGHT |
| ┿ | 253F | BOX DRAWINGS VERTICAL LIGHT AND HORIZONTAL HEAVY |
| ╀ | 2540 | BOX DRAWINGS UP HEAVY AND DOWN HORIZONTAL LIGHT |
| ╁ | 2541 | BOX DRAWINGS DOWN HEAVY AND UP HORIZONTAL LIGHT |
| ╂ | 2542 | BOX DRAWINGS VERTICAL HEAVY AND HORIZONTAL LIGHT |
| ╃ | 2543 | BOX DRAWINGS LEFT UP HEAVY AND RIGHT DOWN LIGHT |
| ╄ | 2544 | BOX DRAWINGS RIGHT UP HEAVY AND LEFT DOWN LIGHT |
| ╅ | 2545 | BOX DRAWINGS LEFT DOWN HEAVY AND RIGHT UP LIGHT |
| ╆ | 2546 | BOX DRAWINGS RIGHT DOWN HEAVY AND LEFT UP LIGHT |
| ╇ | 2547 | BOX DRAWINGS DOWN LIGHT AND UP HORIZONTAL HEAVY |
| ╈ | 2548 | BOX DRAWINGS UP LIGHT AND DOWN HORIZONTAL HEAVY |
| ╉ | 2549 | BOX DRAWINGS RIGHT LIGHT AND LEFT VERTICAL HEAVY |
| ╊ | 254A | BOX DRAWINGS LEFT LIGHT AND RIGHT VERTICAL HEAVY |
| ╋ | 254B | BOX DRAWINGS HEAVY VERTICAL AND HORIZONTAL |
| ╌ | 254C | BOX DRAWINGS LIGHT DOUBLE DASH HORIZONTAL |
| ╍ | 254D | BOX DRAWINGS HEAVY DOUBLE DASH HORIZONTAL |
| ╎ | 254E | BOX DRAWINGS LIGHT DOUBLE DASH VERTICAL |
| ╏ | 254F | BOX DRAWINGS HEAVY DOUBLE DASH VERTICAL |
| ═ | 2550 | BOX DRAWINGS DOUBLE HORIZONTAL |
| ║ | 2551 | BOX DRAWINGS DOUBLE VERTICAL |
| ╒ | 2552 | BOX DRAWINGS DOWN SINGLE AND RIGHT DOUBLE |
| ╓ | 2553 | BOX DRAWINGS DOWN DOUBLE AND RIGHT SINGLE |
| ╔ | 2554 | BOX DRAWINGS DOUBLE DOWN AND RIGHT |
| ╕ | 2555 | BOX DRAWINGS DOWN SINGLE AND LEFT DOUBLE |
| ╖ | 2556 | BOX DRAWINGS DOWN DOUBLE AND LEFT SINGLE |
| ╗ | 2557 | BOX DRAWINGS DOUBLE DOWN AND LEFT |
| ╘ | 2558 | BOX DRAWINGS UP SINGLE AND RIGHT DOUBLE |
| ╙ | 2559 | BOX DRAWINGS UP DOUBLE AND RIGHT SINGLE |
| ╚ | 255A | BOX DRAWINGS DOUBLE UP AND RIGHT |
| ╛ | 255B | BOX DRAWINGS UP SINGLE AND LEFT DOUBLE |
| ╜ | 255C | BOX DRAWINGS UP DOUBLE AND LEFT SINGLE |
| ╝ | 255D | BOX DRAWINGS DOUBLE UP AND LEFT |
| ╞ | 255E | BOX DRAWINGS VERTICAL SINGLE AND RIGHT DOUBLE |
| ╟ | 255F | BOX DRAWINGS VERTICAL DOUBLE AND RIGHT SINGLE |
| ╠ | 2560 | BOX DRAWINGS DOUBLE VERTICAL AND RIGHT |
| ╡ | 2561 | BOX DRAWINGS VERTICAL SINGLE AND LEFT DOUBLE |
| ╢ | 2562 | BOX DRAWINGS VERTICAL DOUBLE AND LEFT SINGLE |
| ╣ | 2563 | BOX DRAWINGS DOUBLE VERTICAL AND LEFT |
| ╤ | 2564 | BOX DRAWINGS DOWN SINGLE AND HORIZONTAL DOUBLE |
| ╥ | 2565 | BOX DRAWINGS DOWN DOUBLE AND HORIZONTAL SINGLE |
| ╦ | 2566 | BOX DRAWINGS DOUBLE DOWN AND HORIZONTAL |
| ╧ | 2567 | BOX DRAWINGS UP SINGLE AND HORIZONTAL DOUBLE |
| ╨ | 2568 | BOX DRAWINGS UP DOUBLE AND HORIZONTAL SINGLE |
| ╩ | 2569 | BOX DRAWINGS DOUBLE UP AND HORIZONTAL |
| ╪ | 256A | BOX DRAWINGS VERTICAL SINGLE AND HORIZONTAL DOUBLE |
| ╫ | 256B | BOX DRAWINGS VERTICAL DOUBLE AND HORIZONTAL SINGLE |
| ╬ | 256C | BOX DRAWINGS DOUBLE VERTICAL AND HORIZONTAL |
| ╭ | 256D | BOX DRAWINGS LIGHT ARC DOWN AND RIGHT |
| ╮ | 256E | BOX DRAWINGS LIGHT ARC DOWN AND LEFT |
| ╯ | 256F | BOX DRAWINGS LIGHT ARC UP AND LEFT |
| ╰ | 2570 | BOX DRAWINGS LIGHT ARC UP AND RIGHT |
| ╱ | 2571 | BOX DRAWINGS LIGHT DIAGONAL UPPER RIGHT TO LOWER LEFT |
| ╲ | 2572 | BOX DRAWINGS LIGHT DIAGONAL UPPER LEFT TO LOWER RIGHT |
| ╳ | 2573 | BOX DRAWINGS LIGHT DIAGONAL CROSS |
| ╴ | 2574 | BOX DRAWINGS LIGHT LEFT |
| ╵ | 2575 | BOX DRAWINGS LIGHT UP |
| ╶ | 2576 | BOX DRAWINGS LIGHT RIGHT |
| ╷ | 2577 | BOX DRAWINGS LIGHT DOWN |
| ╸ | 2578 | BOX DRAWINGS HEAVY LEFT |
| ╹ | 2579 | BOX DRAWINGS HEAVY UP |
| ╺ | 257A | BOX DRAWINGS HEAVY RIGHT |
| ╻ | 257B | BOX DRAWINGS HEAVY DOWN |
| ╼ | 257C | BOX DRAWINGS LIGHT LEFT AND HEAVY RIGHT |
| ╽ | 257D | BOX DRAWINGS LIGHT UP AND HEAVY DOWN |
| ╾ | 257E | BOX DRAWINGS HEAVY LEFT AND LIGHT RIGHT |
| ╿ | 257F | BOX DRAWINGS HEAVY UP AND LIGHT DOWN |
| ▀ | 2580 | UPPER HALF BLOCK |
| ▁ | 2581 | LOWER ONE EIGHTH BLOCK |
| ▂ | 2582 | LOWER ONE QUARTER BLOCK |
| ▃ | 2583 | LOWER THREE EIGHTHS BLOCK |
| ▄ | 2584 | LOWER HALF BLOCK |
| ▅ | 2585 | LOWER FIVE EIGHTHS BLOCK |
| ▆ | 2586 | LOWER THREE QUARTERS BLOCK |
| ▇ | 2587 | LOWER SEVEN EIGHTHS BLOCK |
| █ | 2588 | FULL BLOCK |
| ▉ | 2589 | LEFT SEVEN EIGHTHS BLOCK |
| ▊ | 258A | LEFT THREE QUARTERS BLOCK |
| ▋ | 258B | LEFT FIVE EIGHTHS BLOCK |
| ▌ | 258C | LEFT HALF BLOCK |
| ▍ | 258D | LEFT THREE EIGHTHS BLOCK |
| ▎ | 258E | LEFT ONE QUARTER BLOCK |
| ▏ | 258F | LEFT ONE EIGHTH BLOCK |
| ▐ | 2590 | RIGHT HALF BLOCK |
| ░ | 2591 | LIGHT SHADE |
| ▒ | 2592 | MEDIUM SHADE |
| ▓ | 2593 | DARK SHADE |
| ▔ | 2594 | UPPER ONE EIGHTH BLOCK |
| ▕ | 2595 | RIGHT ONE EIGHTH BLOCK |
| ▖ | 2596 | QUADRANT LOWER LEFT |
| ▗ | 2597 | QUADRANT LOWER RIGHT |
| ▘ | 2598 | QUADRANT UPPER LEFT |
| ▙ | 2599 | QUADRANT UPPER LEFT AND LOWER LEFT AND LOWER RIGHT |
| ▚ | 259A | QUADRANT UPPER LEFT AND LOWER RIGHT |
| ▛ | 259B | QUADRANT UPPER LEFT AND UPPER RIGHT AND LOWER LEFT |
| ▜ | 259C | QUADRANT UPPER LEFT AND UPPER RIGHT AND LOWER RIGHT |
| ▝ | 259D | QUADRANT UPPER RIGHT |
| ▞ | 259E | QUADRANT UPPER RIGHT AND LOWER LEFT |
| ▟ | 259F | QUADRANT UPPER RIGHT AND LOWER LEFT AND LOWER RIGHT |
| ■ | 25A0 | BLACK SQUARE |
| □ | 25A1 | WHITE SQUARE |
| ▢ | 25A2 | WHITE SQUARE WITH ROUNDED CORNERS |
| ▣ | 25A3 | WHITE SQUARE CONTAINING BLACK SMALL SQUARE |
| ▤ | 25A4 | SQUARE WITH HORIZONTAL FILL |
| ▥ | 25A5 | SQUARE WITH VERTICAL FILL |
| ▦ | 25A6 | SQUARE WITH ORTHOGONAL CROSSHATCH FILL |
| ▧ | 25A7 | SQUARE WITH UPPER LEFT TO LOWER RIGHT FILL |
| ▨ | 25A8 | SQUARE WITH UPPER RIGHT TO LOWER LEFT FILL |
| ▩ | 25A9 | SQUARE WITH DIAGONAL CROSSHATCH FILL |
| ▪ | 25AA | BLACK SMALL SQUARE |
| ▫ | 25AB | WHITE SMALL SQUARE |
| ▬ | 25AC | BLACK RECTANGLE |
| ▭ | 25AD | WHITE RECTANGLE |
| ▮ | 25AE | BLACK VERTICAL RECTANGLE |
| ▯ | 25AF | WHITE VERTICAL RECTANGLE |
| ▰ | 25B0 | BLACK PARALLELOGRAM |
| ▱ | 25B1 | WHITE PARALLELOGRAM |
| ▲ | 25B2 | BLACK UP-POINTING TRIANGLE |
| △ | 25B3 | WHITE UP-POINTING TRIANGLE |
| ▴ | 25B4 | BLACK UP-POINTING SMALL TRIANGLE |
| ▵ | 25B5 | WHITE UP-POINTING SMALL TRIANGLE |
| ▶ | 25B6 | BLACK RIGHT-POINTING TRIANGLE |
| ▷ | 25B7 | WHITE RIGHT-POINTING TRIANGLE |
| ▸ | 25B8 | BLACK RIGHT-POINTING SMALL TRIANGLE |
| ▹ | 25B9 | WHITE RIGHT-POINTING SMALL TRIANGLE |
| ► | 25BA | BLACK RIGHT-POINTING POINTER |
| ▻ | 25BB | WHITE RIGHT-POINTING POINTER |
| ▼ | 25BC | BLACK DOWN-POINTING TRIANGLE |
| ▽ | 25BD | WHITE DOWN-POINTING TRIANGLE |
| ▾ | 25BE | BLACK DOWN-POINTING SMALL TRIANGLE |
| ▿ | 25BF | WHITE DOWN-POINTING SMALL TRIANGLE |
| ◀ | 25C0 | BLACK LEFT-POINTING TRIANGLE |
| ◁ | 25C1 | WHITE LEFT-POINTING TRIANGLE |
| ◂ | 25C2 | BLACK LEFT-POINTING SMALL TRIANGLE |
| ◃ | 25C3 | WHITE LEFT-POINTING SMALL TRIANGLE |
| ◄ | 25C4 | BLACK LEFT-POINTING POINTER |
| ◅ | 25C5 | WHITE LEFT-POINTING POINTER |
| ◆ | 25C6 | BLACK DIAMOND |
| ◇ | 25C7 | WHITE DIAMOND |
| ◈ | 25C8 | WHITE DIAMOND CONTAINING BLACK SMALL DIAMOND |
| ◉ | 25C9 | FISHEYE |
| ◊ | 25CA | LOZENGE |
| ○ | 25CB | WHITE CIRCLE |
| ◌ | 25CC | DOTTED CIRCLE |
| ◍ | 25CD | CIRCLE WITH VERTICAL FILL |
| ◎ | 25CE | BULLSEYE |
| ● | 25CF | BLACK CIRCLE |
| ◐ | 25D0 | CIRCLE WITH LEFT HALF BLACK |
| ◑ | 25D1 | CIRCLE WITH RIGHT HALF BLACK |
| ◒ | 25D2 | CIRCLE WITH LOWER HALF BLACK |
| ◓ | 25D3 | CIRCLE WITH UPPER HALF BLACK |
| ◔ | 25D4 | CIRCLE WITH UPPER RIGHT QUADRANT BLACK |
| ◕ | 25D5 | CIRCLE WITH ALL BUT UPPER LEFT QUADRANT BLACK |
| ◖ | 25D6 | LEFT HALF BLACK CIRCLE |
| ◗ | 25D7 | RIGHT HALF BLACK CIRCLE |
| ◘ | 25D8 | INVERSE BULLET |
| ◙ | 25D9 | INVERSE WHITE CIRCLE |
| ◚ | 25DA | UPPER HALF INVERSE WHITE CIRCLE |
| ◛ | 25DB | LOWER HALF INVERSE WHITE CIRCLE |
| ◜ | 25DC | UPPER LEFT QUADRANT CIRCULAR ARC |
| ◝ | 25DD | UPPER RIGHT QUADRANT CIRCULAR ARC |
| ◞ | 25DE | LOWER RIGHT QUADRANT CIRCULAR ARC |
| ◟ | 25DF | LOWER LEFT QUADRANT CIRCULAR ARC |
| ◠ | 25E0 | UPPER HALF CIRCLE |
| ◡ | 25E1 | LOWER HALF CIRCLE |
| ◢ | 25E2 | BLACK LOWER RIGHT TRIANGLE |
| ◣ | 25E3 | BLACK LOWER LEFT TRIANGLE |
| ◤ | 25E4 | BLACK UPPER LEFT TRIANGLE |
| ◥ | 25E5 | BLACK UPPER RIGHT TRIANGLE |
| ◦ | 25E6 | WHITE BULLET |
| ◧ | 25E7 | SQUARE WITH LEFT HALF BLACK |
| ◨ | 25E8 | SQUARE WITH RIGHT HALF BLACK |
| ◩ | 25E9 | SQUARE WITH UPPER LEFT DIAGONAL HALF BLACK |
| ◪ | 25EA | SQUARE WITH LOWER RIGHT DIAGONAL HALF BLACK |
| ◫ | 25EB | WHITE SQUARE WITH VERTICAL BISECTING LINE |
| ◬ | 25EC | WHITE UP-POINTING TRIANGLE WITH DOT |
| ◭ | 25ED | UP-POINTING TRIANGLE WITH LEFT HALF BLACK |
| ◮ | 25EE | UP-POINTING TRIANGLE WITH RIGHT HALF BLACK |
| ◯ | 25EF | LARGE CIRCLE |
| ◰ | 25F0 | WHITE SQUARE WITH UPPER LEFT QUADRANT |
| ◱ | 25F1 | WHITE SQUARE WITH LOWER LEFT QUADRANT |
| ◲ | 25F2 | WHITE SQUARE WITH LOWER RIGHT QUADRANT |
| ◳ | 25F3 | WHITE SQUARE WITH UPPER RIGHT QUADRANT |
| ◴ | 25F4 | WHITE CIRCLE WITH UPPER LEFT QUADRANT |
| ◵ | 25F5 | WHITE CIRCLE WITH LOWER LEFT QUADRANT |
| ◶ | 25F6 | WHITE CIRCLE WITH LOWER RIGHT QUADRANT |
| ◷ | 25F7 | WHITE CIRCLE WITH UPPER RIGHT QUADRANT |
| ◸ | 25F8 | UPPER LEFT TRIANGLE |
| ◹ | 25F9 | UPPER RIGHT TRIANGLE |
| ◺ | 25FA | LOWER LEFT TRIANGLE |
| ◻ | 25FB | WHITE MEDIUM SQUARE |
| ◼ | 25FC | BLACK MEDIUM SQUARE |
| ◽ | 25FD | WHITE MEDIUM SMALL SQUARE |
| ◾ | 25FE | BLACK MEDIUM SMALL SQUARE |
| ◿ | 25FF | LOWER RIGHT TRIANGLE |
| ☀ | 2600 | BLACK SUN WITH RAYS |
| ☁ | 2601 | CLOUD |
| ☂ | 2602 | UMBRELLA |
| ☃ | 2603 | SNOWMAN |
| ☄ | 2604 | COMET |
| ★ | 2605 | BLACK STAR |
| ☆ | 2606 | WHITE STAR |
| ☇ | 2607 | LIGHTNING |
| ☈ | 2608 | THUNDERSTORM |
| ☉ | 2609 | SUN |
| ☊ | 260A | ASCENDING NODE |
| ☋ | 260B | DESCENDING NODE |
| ☌ | 260C | CONJUNCTION |
| ☍ | 260D | OPPOSITION |
| ☎ | 260E | BLACK TELEPHONE |
| ☏ | 260F | WHITE TELEPHONE |
| ☐ | 2610 | BALLOT BOX |
| ☑ | 2611 | BALLOT BOX WITH CHECK |
| ☒ | 2612 | BALLOT BOX WITH X |
| ☓ | 2613 | SALTIRE |
| ☔ | 2614 | UMBRELLA WITH RAIN DROPS |
| ☕ | 2615 | HOT BEVERAGE |
| ☖ | 2616 | WHITE SHOGI PIECE |
| ☗ | 2617 | BLACK SHOGI PIECE |
| ☘ | 2618 | SHAMROCK |
| ☙ | 2619 | REVERSED ROTATED FLORAL HEART BULLET |
| ☚ | 261A | BLACK LEFT POINTING INDEX |
| ☛ | 261B | BLACK RIGHT POINTING INDEX |
| ☜ | 261C | WHITE LEFT POINTING INDEX |
| ☝ | 261D | WHITE UP POINTING INDEX |
| ☞ | 261E | WHITE RIGHT POINTING INDEX |
| ☟ | 261F | WHITE DOWN POINTING INDEX |
| ☠ | 2620 | SKULL AND CROSSBONES |
| ☡ | 2621 | CAUTION SIGN |
| ☢ | 2622 | RADIOACTIVE SIGN |
| ☣ | 2623 | BIOHAZARD SIGN |
| ☤ | 2624 | CADUCEUS |
| ☥ | 2625 | ANKH |
| ☦ | 2626 | ORTHODOX CROSS |
| ☧ | 2627 | CHI RHO |
| ☨ | 2628 | CROSS OF LORRAINE |
| ☩ | 2629 | CROSS OF JERUSALEM |
| ☪ | 262A | STAR AND CRESCENT |
| ☫ | 262B | FARSI SYMBOL |
| ☬ | 262C | KHANDA |
| ☭ | 262D | HAMMER AND SICKLE |
| ☮ | 262E | PEACE SYMBOL |
| ☯ | 262F | YIN YANG |
| ☰ | 2630 | TRIGRAM FOR HEAVEN |
| ☱ | 2631 | TRIGRAM FOR LAKE |
| ☲ | 2632 | TRIGRAM FOR FIRE |
| ☳ | 2633 | TRIGRAM FOR THUNDER |
| ☴ | 2634 | TRIGRAM FOR WIND |
| ☵ | 2635 | TRIGRAM FOR WATER |
| ☶ | 2636 | TRIGRAM FOR MOUNTAIN |
| ☷ | 2637 | TRIGRAM FOR EARTH |
| ☸ | 2638 | WHEEL OF DHARMA |
| ☹ | 2639 | WHITE FROWNING FACE |
| ☺ | 263A | WHITE SMILING FACE |
| ☻ | 263B | BLACK SMILING FACE |
| ☼ | 263C | WHITE SUN WITH RAYS |
| ☽ | 263D | FIRST QUARTER MOON |
| ☾ | 263E | LAST QUARTER MOON |
| ☿ | 263F | MERCURY |
| ♀ | 2640 | FEMALE SIGN |
| ♁ | 2641 | EARTH |
| ♂ | 2642 | MALE SIGN |
| ♃ | 2643 | JUPITER |
| ♄ | 2644 | SATURN |
| ♅ | 2645 | URANUS |
| ♆ | 2646 | NEPTUNE |
| ♇ | 2647 | PLUTO |
| ♈ | 2648 | ARIES |
| ♉ | 2649 | TAURUS |
| ♊ | 264A | GEMINI |
| ♋ | 264B | CANCER |
| ♌ | 264C | LEO |
| ♍ | 264D | VIRGO |
| ♎ | 264E | LIBRA |
| ♏ | 264F | SCORPIUS |
| ♐ | 2650 | SAGITTARIUS |
| ♑ | 2651 | CAPRICORN |
| ♒ | 2652 | AQUARIUS |
| ♓ | 2653 | PISCES |
| ♔ | 2654 | WHITE CHESS KING |
| ♕ | 2655 | WHITE CHESS QUEEN |
| ♖ | 2656 | WHITE CHESS ROOK |
| ♗ | 2657 | WHITE CHESS BISHOP |
| ♘ | 2658 | WHITE CHESS KNIGHT |
| ♙ | 2659 | WHITE CHESS PAWN |
| ♚ | 265A | BLACK CHESS KING |
| ♛ | 265B | BLACK CHESS QUEEN |
| ♜ | 265C | BLACK CHESS ROOK |
| ♝ | 265D | BLACK CHESS BISHOP |
| ♞ | 265E | BLACK CHESS KNIGHT |
| ♟ | 265F | BLACK CHESS PAWN |
| ♠ | 2660 | BLACK SPADE SUIT |
| ♡ | 2661 | WHITE HEART SUIT |
| ♢ | 2662 | WHITE DIAMOND SUIT |
| ♣ | 2663 | BLACK CLUB SUIT |
| ♤ | 2664 | WHITE SPADE SUIT |
| ♥ | 2665 | BLACK HEART SUIT |
| ♦ | 2666 | BLACK DIAMOND SUIT |
| ♧ | 2667 | WHITE CLUB SUIT |
| ♨ | 2668 | HOT SPRINGS |
| ♩ | 2669 | QUARTER NOTE |
| ♪ | 266A | EIGHTH NOTE |
| ♫ | 266B | BEAMED EIGHTH NOTES |
| ♬ | 266C | BEAMED SIXTEENTH NOTES |
| ♭ | 266D | MUSIC FLAT SIGN |
| ♮ | 266E | MUSIC NATURAL SIGN |
| ♯ | 266F | MUSIC SHARP SIGN |
| ♰ | 2670 | WEST SYRIAC CROSS |
| ♱ | 2671 | EAST SYRIAC CROSS |
| ♲ | 2672 | UNIVERSAL RECYCLING SYMBOL |
| ♳ | 2673 | RECYCLING SYMBOL FOR TYPE-1 PLASTICS |
| ♴ | 2674 | RECYCLING SYMBOL FOR TYPE-2 PLASTICS |
| ♵ | 2675 | RECYCLING SYMBOL FOR TYPE-3 PLASTICS |
| ♶ | 2676 | RECYCLING SYMBOL FOR TYPE-4 PLASTICS |
| ♷ | 2677 | RECYCLING SYMBOL FOR TYPE-5 PLASTICS |
| ♸ | 2678 | RECYCLING SYMBOL FOR TYPE-6 PLASTICS |
| ♹ | 2679 | RECYCLING SYMBOL FOR TYPE-7 PLASTICS |
| ♺ | 267A | RECYCLING SYMBOL FOR GENERIC MATERIALS |
| ♻ | 267B | BLACK UNIVERSAL RECYCLING SYMBOL |
| ♼ | 267C | RECYCLED PAPER SYMBOL |
| ♽ | 267D | PARTIALLY-RECYCLED PAPER SYMBOL |
| ♾ | 267E | PERMANENT PAPER SIGN |
| ♿ | 267F | WHEELCHAIR SYMBOL |
| ⚀ | 2680 | DIE FACE-1 |
| ⚁ | 2681 | DIE FACE-2 |
| ⚂ | 2682 | DIE FACE-3 |
| ⚃ | 2683 | DIE FACE-4 |
| ⚄ | 2684 | DIE FACE-5 |
| ⚅ | 2685 | DIE FACE-6 |
| ⚆ | 2686 | WHITE CIRCLE WITH DOT RIGHT |
| ⚇ | 2687 | WHITE CIRCLE WITH TWO DOTS |
| ⚈ | 2688 | BLACK CIRCLE WITH WHITE DOT RIGHT |
| ⚉ | 2689 | BLACK CIRCLE WITH TWO WHITE DOTS |
| ⚊ | 268A | MONOGRAM FOR YANG |
| ⚋ | 268B | MONOGRAM FOR YIN |
| ⚌ | 268C | DIGRAM FOR GREATER YANG |
| ⚍ | 268D | DIGRAM FOR LESSER YIN |
| ⚎ | 268E | DIGRAM FOR LESSER YANG |
| ⚏ | 268F | DIGRAM FOR GREATER YIN |
| ⚐ | 2690 | WHITE FLAG |
| ⚑ | 2691 | BLACK FLAG |
| ⚒ | 2692 | HAMMER AND PICK |
| ⚓ | 2693 | ANCHOR |
| ⚔ | 2694 | CROSSED SWORDS |
| ⚕ | 2695 | STAFF OF AESCULAPIUS |
| ⚖ | 2696 | SCALES |
| ⚗ | 2697 | ALEMBIC |
| ⚘ | 2698 | FLOWER |
| ⚙ | 2699 | GEAR |
| ⚚ | 269A | STAFF OF HERMES |
| ⚛ | 269B | ATOM SYMBOL |
| ⚜ | 269C | FLEUR-DE-LIS |
| ⚝ | 269D | OUTLINED WHITE STAR |
| ⚞ | 269E | THREE LINES CONVERGING RIGHT |
| ⚟ | 269F | THREE LINES CONVERGING LEFT |
| ⚠ | 26A0 | WARNING SIGN |
| ⚡ | 26A1 | HIGH VOLTAGE SIGN |
| ⚢ | 26A2 | DOUBLED FEMALE SIGN |
| ⚣ | 26A3 | DOUBLED MALE SIGN |
| ⚤ | 26A4 | INTERLOCKED FEMALE AND MALE SIGN |
| ⚥ | 26A5 | MALE AND FEMALE SIGN |
| ⚦ | 26A6 | MALE WITH STROKE SIGN |
| ⚧ | 26A7 | MALE WITH STROKE AND MALE AND FEMALE SIGN |
| ⚨ | 26A8 | VERTICAL MALE WITH STROKE SIGN |
| ⚩ | 26A9 | HORIZONTAL MALE WITH STROKE SIGN |
| ⚪ | 26AA | MEDIUM WHITE CIRCLE |
| ⚫ | 26AB | MEDIUM BLACK CIRCLE |
| ⚬ | 26AC | MEDIUM SMALL WHITE CIRCLE |
| ⚭ | 26AD | MARRIAGE SYMBOL |
| ⚮ | 26AE | DIVORCE SYMBOL |
| ⚯ | 26AF | UNMARRIED PARTNERSHIP SYMBOL |
| ⚰ | 26B0 | COFFIN |
| ⚱ | 26B1 | FUNERAL URN |
| ⚲ | 26B2 | NEUTER |
| ⚳ | 26B3 | CERES |
| ⚴ | 26B4 | PALLAS |
| ⚵ | 26B5 | JUNO |
| ⚶ | 26B6 | VESTA |
| ⚷ | 26B7 | CHIRON |
| ⚸ | 26B8 | BLACK MOON LILITH |
| ⚹ | 26B9 | SEXTILE |
| ⚺ | 26BA | SEMISEXTILE |
| ⚻ | 26BB | QUINCUNX |
| ⚼ | 26BC | SESQUIQUADRATE |
| ⚽ | 26BD | SOCCER BALL |
| ⚾ | 26BE | BASEBALL |
| ⚿ | 26BF | SQUARED KEY |
| ⛀ | 26C0 | WHITE DRAUGHTS MAN |
| ⛁ | 26C1 | WHITE DRAUGHTS KING |
| ⛂ | 26C2 | BLACK DRAUGHTS MAN |
| ⛃ | 26C3 | BLACK DRAUGHTS KING |
| ⛄ | 26C4 | SNOWMAN WITHOUT SNOW |
| ⛅ | 26C5 | SUN BEHIND CLOUD |
| ⛆ | 26C6 | RAIN |
| ⛇ | 26C7 | BLACK SNOWMAN |
| ⛈ | 26C8 | THUNDER CLOUD AND RAIN |
| ⛉ | 26C9 | TURNED WHITE SHOGI PIECE |
| ⛊ | 26CA | TURNED BLACK SHOGI PIECE |
| ⛋ | 26CB | WHITE DIAMOND IN SQUARE |
| ⛌ | 26CC | CROSSING LANES |
| ⛍ | 26CD | DISABLED CAR |
| ⛎ | 26CE | OPHIUCHUS |
| ⛏ | 26CF | PICK |
| ⛐ | 26D0 | CAR SLIDING |
| ⛑ | 26D1 | HELMET WITH WHITE CROSS |
| ⛒ | 26D2 | CIRCLED CROSSING LANES |
| ⛓ | 26D3 | CHAINS |
| ⛔ | 26D4 | NO ENTRY |
| ⛕ | 26D5 | ALTERNATE ONE-WAY LEFT WAY TRAFFIC |
| ⛖ | 26D6 | BLACK TWO-WAY LEFT WAY TRAFFIC |
| ⛗ | 26D7 | WHITE TWO-WAY LEFT WAY TRAFFIC |
| ⛘ | 26D8 | BLACK LEFT LANE MERGE |
| ⛙ | 26D9 | WHITE LEFT LANE MERGE |
| ⛚ | 26DA | DRIVE SLOW SIGN |
| ⛛ | 26DB | HEAVY WHITE DOWN-POINTING TRIANGLE |
| ⛜ | 26DC | LEFT CLOSED ENTRY |
| ⛝ | 26DD | SQUARED SALTIRE |
| ⛞ | 26DE | FALLING DIAGONAL IN WHITE CIRCLE IN BLACK SQUARE |
| ⛟ | 26DF | BLACK TRUCK |
| ⛠ | 26E0 | RESTRICTED LEFT ENTRY-1 |
| ⛡ | 26E1 | RESTRICTED LEFT ENTRY-2 |
| ⛢ | 26E2 | ASTRONOMICAL SYMBOL FOR URANUS |
| ⛣ | 26E3 | HEAVY CIRCLE WITH STROKE AND TWO DOTS ABOVE |
| ⛤ | 26E4 | PENTAGRAM |
| ⛥ | 26E5 | RIGHT-HANDED INTERLACED PENTAGRAM |
| ⛦ | 26E6 | LEFT-HANDED INTERLACED PENTAGRAM |
| ⛧ | 26E7 | INVERTED PENTAGRAM |
| ⛨ | 26E8 | BLACK CROSS ON SHIELD |
| ⛩ | 26E9 | SHINTO SHRINE |
| ⛪ | 26EA | CHURCH |
| ⛫ | 26EB | CASTLE |
| ⛬ | 26EC | HISTORIC SITE |
| ⛭ | 26ED | GEAR WITHOUT HUB |
| ⛮ | 26EE | GEAR WITH HANDLES |
| ⛯ | 26EF | MAP SYMBOL FOR LIGHTHOUSE |
| ⛰ | 26F0 | MOUNTAIN |
| ⛱ | 26F1 | UMBRELLA ON GROUND |
| ⛲ | 26F2 | FOUNTAIN |
| ⛳ | 26F3 | FLAG IN HOLE |
| ⛴ | 26F4 | FERRY |
| ⛵ | 26F5 | SAILBOAT |
| ⛶ | 26F6 | SQUARE FOUR CORNERS |
| ⛷ | 26F7 | SKIER |
| ⛸ | 26F8 | ICE SKATE |
| ⛹ | 26F9 | PERSON WITH BALL |
| ⛺ | 26FA | TENT |
| ⛻ | 26FB | JAPANESE BANK SYMBOL |
| ⛼ | 26FC | HEADSTONE GRAVEYARD SYMBOL |
| ⛽ | 26FD | FUEL PUMP |
| ⛾ | 26FE | CUP ON BLACK SQUARE |
| ⛿ | 26FF | WHITE FLAG WITH HORIZONTAL MIDDLE BLACK STRIPE |
| ✁ | 2701 | UPPER BLADE SCISSORS |
| ✂ | 2702 | BLACK SCISSORS |
| ✃ | 2703 | LOWER BLADE SCISSORS |
| ✄ | 2704 | WHITE SCISSORS |
| ✅ | 2705 | WHITE HEAVY CHECK MARK |
| ✆ | 2706 | TELEPHONE LOCATION SIGN |
| ✇ | 2707 | TAPE DRIVE |
| ✈ | 2708 | AIRPLANE |
| ✉ | 2709 | ENVELOPE |
| ✊ | 270A | RAISED FIST |
| ✋ | 270B | RAISED HAND |
| ✌ | 270C | VICTORY HAND |
| ✍ | 270D | WRITING HAND |
| ✎ | 270E | LOWER RIGHT PENCIL |
| ✏ | 270F | PENCIL |
| ✐ | 2710 | UPPER RIGHT PENCIL |
| ✑ | 2711 | WHITE NIB |
| ✒ | 2712 | BLACK NIB |
| ✓ | 2713 | CHECK MARK |
| ✔ | 2714 | HEAVY CHECK MARK |
| ✕ | 2715 | MULTIPLICATION X |
| ✖ | 2716 | HEAVY MULTIPLICATION X |
| ✗ | 2717 | BALLOT X |
| ✘ | 2718 | HEAVY BALLOT X |
| ✙ | 2719 | OUTLINED GREEK CROSS |
| ✚ | 271A | HEAVY GREEK CROSS |
| ✛ | 271B | OPEN CENTRE CROSS |
| ✜ | 271C | HEAVY OPEN CENTRE CROSS |
| ✝ | 271D | LATIN CROSS |
| ✞ | 271E | SHADOWED WHITE LATIN CROSS |
| ✟ | 271F | OUTLINED LATIN CROSS |
| ✠ | 2720 | MALTESE CROSS |
| ✡ | 2721 | STAR OF DAVID |
| ✢ | 2722 | FOUR TEARDROP-SPOKED ASTERISK |
| ✣ | 2723 | FOUR BALLOON-SPOKED ASTERISK |
| ✤ | 2724 | HEAVY FOUR BALLOON-SPOKED ASTERISK |
| ✥ | 2725 | FOUR CLUB-SPOKED ASTERISK |
| ✦ | 2726 | BLACK FOUR POINTED STAR |
| ✧ | 2727 | WHITE FOUR POINTED STAR |
| ✨ | 2728 | SPARKLES |
| ✩ | 2729 | STRESS OUTLINED WHITE STAR |
| ✪ | 272A | CIRCLED WHITE STAR |
| ✫ | 272B | OPEN CENTRE BLACK STAR |
| ✬ | 272C | BLACK CENTRE WHITE STAR |
| ✭ | 272D | OUTLINED BLACK STAR |
| ✮ | 272E | HEAVY OUTLINED BLACK STAR |
| ✯ | 272F | PINWHEEL STAR |
| ✰ | 2730 | SHADOWED WHITE STAR |
| ✱ | 2731 | HEAVY ASTERISK |
| ✲ | 2732 | OPEN CENTRE ASTERISK |
| ✳ | 2733 | EIGHT SPOKED ASTERISK |
| ✴ | 2734 | EIGHT POINTED BLACK STAR |
| ✵ | 2735 | EIGHT POINTED PINWHEEL STAR |
| ✶ | 2736 | SIX POINTED BLACK STAR |
| ✷ | 2737 | EIGHT POINTED RECTILINEAR BLACK STAR |
| ✸ | 2738 | HEAVY EIGHT POINTED RECTILINEAR BLACK STAR |
| ✹ | 2739 | TWELVE POINTED BLACK STAR |
| ✺ | 273A | SIXTEEN POINTED ASTERISK |
| ✻ | 273B | TEARDROP-SPOKED ASTERISK |
| ✼ | 273C | OPEN CENTRE TEARDROP-SPOKED ASTERISK |
| ✽ | 273D | HEAVY TEARDROP-SPOKED ASTERISK |
| ✾ | 273E | SIX PETALLED BLACK AND WHITE FLORETTE |
| ✿ | 273F | BLACK FLORETTE |
| ❀ | 2740 | WHITE FLORETTE |
| ❁ | 2741 | EIGHT PETALLED OUTLINED BLACK FLORETTE |
| ❂ | 2742 | CIRCLED OPEN CENTRE EIGHT POINTED STAR |
| ❃ | 2743 | HEAVY TEARDROP-SPOKED PINWHEEL ASTERISK |
| ❄ | 2744 | SNOWFLAKE |
| ❅ | 2745 | TIGHT TRIFOLIATE SNOWFLAKE |
| ❆ | 2746 | HEAVY CHEVRON SNOWFLAKE |
| ❇ | 2747 | SPARKLE |
| ❈ | 2748 | HEAVY SPARKLE |
| ❉ | 2749 | BALLOON-SPOKED ASTERISK |
| ❊ | 274A | EIGHT TEARDROP-SPOKED PROPELLER ASTERISK |
| ❋ | 274B | HEAVY EIGHT TEARDROP-SPOKED PROPELLER ASTERISK |
| ❌ | 274C | CROSS MARK |
| ❍ | 274D | SHADOWED WHITE CIRCLE |
| ❎ | 274E | NEGATIVE SQUARED CROSS MARK |
| ❏ | 274F | LOWER RIGHT DROP-SHADOWED WHITE SQUARE |
| ❐ | 2750 | UPPER RIGHT DROP-SHADOWED WHITE SQUARE |
| ❑ | 2751 | LOWER RIGHT SHADOWED WHITE SQUARE |
| ❒ | 2752 | UPPER RIGHT SHADOWED WHITE SQUARE |
| ❓ | 2753 | BLACK QUESTION MARK ORNAMENT |
| ❔ | 2754 | WHITE QUESTION MARK ORNAMENT |
| ❕ | 2755 | WHITE EXCLAMATION MARK ORNAMENT |
| ❖ | 2756 | BLACK DIAMOND MINUS WHITE X |
| ❗ | 2757 | HEAVY EXCLAMATION MARK SYMBOL |
| ❘ | 2758 | LIGHT VERTICAL BAR |
| ❙ | 2759 | MEDIUM VERTICAL BAR |
| ❚ | 275A | HEAVY VERTICAL BAR |
| ❛ | 275B | HEAVY SINGLE TURNED COMMA QUOTATION MARK ORNAMENT |
| ❜ | 275C | HEAVY SINGLE COMMA QUOTATION MARK ORNAMENT |
| ❝ | 275D | HEAVY DOUBLE TURNED COMMA QUOTATION MARK ORNAMENT |
| ❞ | 275E | HEAVY DOUBLE COMMA QUOTATION MARK ORNAMENT |
| ❟ | 275F | HEAVY LOW SINGLE COMMA QUOTATION MARK ORNAMENT |
| ❠ | 2760 | HEAVY LOW DOUBLE COMMA QUOTATION MARK ORNAMENT |
| ❡ | 2761 | CURVED STEM PARAGRAPH SIGN ORNAMENT |
| ❢ | 2762 | HEAVY EXCLAMATION MARK ORNAMENT |
| ❣ | 2763 | HEAVY HEART EXCLAMATION MARK ORNAMENT |
| ❤ | 2764 | HEAVY BLACK HEART |
| ❥ | 2765 | ROTATED HEAVY BLACK HEART BULLET |
| ❦ | 2766 | FLORAL HEART |
| ❧ | 2767 | ROTATED FLORAL HEART BULLET |
| ❨ | 2768 | MEDIUM LEFT PARENTHESIS ORNAMENT |
| ❩ | 2769 | MEDIUM RIGHT PARENTHESIS ORNAMENT |
| ❪ | 276A | MEDIUM FLATTENED LEFT PARENTHESIS ORNAMENT |
| ❫ | 276B | MEDIUM FLATTENED RIGHT PARENTHESIS ORNAMENT |
| ❬ | 276C | MEDIUM LEFT-POINTING ANGLE BRACKET ORNAMENT |
| ❭ | 276D | MEDIUM RIGHT-POINTING ANGLE BRACKET ORNAMENT |
| ❮ | 276E | HEAVY LEFT-POINTING ANGLE QUOTATION MARK ORNAMENT |
| ❯ | 276F | HEAVY RIGHT-POINTING ANGLE QUOTATION MARK ORNAMENT |
| ❰ | 2770 | HEAVY LEFT-POINTING ANGLE BRACKET ORNAMENT |
| ❱ | 2771 | HEAVY RIGHT-POINTING ANGLE BRACKET ORNAMENT |
| ❲ | 2772 | LIGHT LEFT TORTOISE SHELL BRACKET ORNAMENT |
| ❳ | 2773 | LIGHT RIGHT TORTOISE SHELL BRACKET ORNAMENT |
| ❴ | 2774 | MEDIUM LEFT CURLY BRACKET ORNAMENT |
| ❵ | 2775 | MEDIUM RIGHT CURLY BRACKET ORNAMENT |
| ❶ | 2776 | DINGBAT NEGATIVE CIRCLED DIGIT ONE |
| ❷ | 2777 | DINGBAT NEGATIVE CIRCLED DIGIT TWO |
| ❸ | 2778 | DINGBAT NEGATIVE CIRCLED DIGIT THREE |
| ❹ | 2779 | DINGBAT NEGATIVE CIRCLED DIGIT FOUR |
| ❺ | 277A | DINGBAT NEGATIVE CIRCLED DIGIT FIVE |
| ❻ | 277B | DINGBAT NEGATIVE CIRCLED DIGIT SIX |
| ❼ | 277C | DINGBAT NEGATIVE CIRCLED DIGIT SEVEN |
| ❽ | 277D | DINGBAT NEGATIVE CIRCLED DIGIT EIGHT |
| ❾ | 277E | DINGBAT NEGATIVE CIRCLED DIGIT NINE |
| ❿ | 277F | DINGBAT NEGATIVE CIRCLED NUMBER TEN |
| ➀ | 2780 | DINGBAT CIRCLED SANS-SERIF DIGIT ONE |
| ➁ | 2781 | DINGBAT CIRCLED SANS-SERIF DIGIT TWO |
| ➂ | 2782 | DINGBAT CIRCLED SANS-SERIF DIGIT THREE |
| ➃ | 2783 | DINGBAT CIRCLED SANS-SERIF DIGIT FOUR |
| ➄ | 2784 | DINGBAT CIRCLED SANS-SERIF DIGIT FIVE |
| ➅ | 2785 | DINGBAT CIRCLED SANS-SERIF DIGIT SIX |
| ➆ | 2786 | DINGBAT CIRCLED SANS-SERIF DIGIT SEVEN |
| ➇ | 2787 | DINGBAT CIRCLED SANS-SERIF DIGIT EIGHT |
| ➈ | 2788 | DINGBAT CIRCLED SANS-SERIF DIGIT NINE |
| ➉ | 2789 | DINGBAT CIRCLED SANS-SERIF NUMBER TEN |
| ➊ | 278A | DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT ONE |
| ➋ | 278B | DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT TWO |
| ➌ | 278C | DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT THREE |
| ➍ | 278D | DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT FOUR |
| ➎ | 278E | DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT FIVE |
| ➏ | 278F | DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT SIX |
| ➐ | 2790 | DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT SEVEN |
| ➑ | 2791 | DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT EIGHT |
| ➒ | 2792 | DINGBAT NEGATIVE CIRCLED SANS-SERIF DIGIT NINE |
| ➓ | 2793 | DINGBAT NEGATIVE CIRCLED SANS-SERIF NUMBER TEN |
| ➔ | 2794 | HEAVY WIDE-HEADED RIGHTWARDS ARROW |
| ➕ | 2795 | HEAVY PLUS SIGN |
| ➖ | 2796 | HEAVY MINUS SIGN |
| ➗ | 2797 | HEAVY DIVISION SIGN |
| ➘ | 2798 | HEAVY SOUTH EAST ARROW |
| ➙ | 2799 | HEAVY RIGHTWARDS ARROW |
| ➚ | 279A | HEAVY NORTH EAST ARROW |
| ➛ | 279B | DRAFTING POINT RIGHTWARDS ARROW |
| ➜ | 279C | HEAVY ROUND-TIPPED RIGHTWARDS ARROW |
| ➝ | 279D | TRIANGLE-HEADED RIGHTWARDS ARROW |
| ➞ | 279E | HEAVY TRIANGLE-HEADED RIGHTWARDS ARROW |
| ➟ | 279F | DASHED TRIANGLE-HEADED RIGHTWARDS ARROW |
| ➠ | 27A0 | HEAVY DASHED TRIANGLE-HEADED RIGHTWARDS ARROW |
| ➡ | 27A1 | BLACK RIGHTWARDS ARROW |
| ➢ | 27A2 | THREE-D TOP-LIGHTED RIGHTWARDS ARROWHEAD |
| ➣ | 27A3 | THREE-D BOTTOM-LIGHTED RIGHTWARDS ARROWHEAD |
| ➤ | 27A4 | BLACK RIGHTWARDS ARROWHEAD |
| ➥ | 27A5 | HEAVY BLACK CURVED DOWNWARDS AND RIGHTWARDS ARROW |
| ➦ | 27A6 | HEAVY BLACK CURVED UPWARDS AND RIGHTWARDS ARROW |
| ➧ | 27A7 | SQUAT BLACK RIGHTWARDS ARROW |
| ➨ | 27A8 | HEAVY CONCAVE-POINTED BLACK RIGHTWARDS ARROW |
| ➩ | 27A9 | RIGHT-SHADED WHITE RIGHTWARDS ARROW |
| ➪ | 27AA | LEFT-SHADED WHITE RIGHTWARDS ARROW |
| ➫ | 27AB | BACK-TILTED SHADOWED WHITE RIGHTWARDS ARROW |
| ➬ | 27AC | FRONT-TILTED SHADOWED WHITE RIGHTWARDS ARROW |
| ➭ | 27AD | HEAVY LOWER RIGHT-SHADOWED WHITE RIGHTWARDS ARROW |
| ➮ | 27AE | HEAVY UPPER RIGHT-SHADOWED WHITE RIGHTWARDS ARROW |
| ➯ | 27AF | NOTCHED LOWER RIGHT-SHADOWED WHITE RIGHTWARDS ARROW |
| ➰ | 27B0 | CURLY LOOP |
| ➱ | 27B1 | NOTCHED UPPER RIGHT-SHADOWED WHITE RIGHTWARDS ARROW |
| ➲ | 27B2 | CIRCLED HEAVY WHITE RIGHTWARDS ARROW |
| ➳ | 27B3 | WHITE-FEATHERED RIGHTWARDS ARROW |
| ➴ | 27B4 | BLACK-FEATHERED SOUTH EAST ARROW |
| ➵ | 27B5 | BLACK-FEATHERED RIGHTWARDS ARROW |
| ➶ | 27B6 | BLACK-FEATHERED NORTH EAST ARROW |
| ➷ | 27B7 | HEAVY BLACK-FEATHERED SOUTH EAST ARROW |
| ➸ | 27B8 | HEAVY BLACK-FEATHERED RIGHTWARDS ARROW |
| ➹ | 27B9 | HEAVY BLACK-FEATHERED NORTH EAST ARROW |
| ➺ | 27BA | TEARDROP-BARBED RIGHTWARDS ARROW |
| ➻ | 27BB | HEAVY TEARDROP-SHANKED RIGHTWARDS ARROW |
| ➼ | 27BC | WEDGE-TAILED RIGHTWARDS ARROW |
| ➽ | 27BD | HEAVY WEDGE-TAILED RIGHTWARDS ARROW |
| ➾ | 27BE | OPEN-OUTLINED RIGHTWARDS ARROW |
| ➿ | 27BF | DOUBLE CURLY LOOP |

PHỤ LỤC

Tai lieu tham khao

THAM CHIẾU CHỨC NĂNG CSS

Các hàm CSS được sử dụng làm giá trị cho các thuộc tính CSS khác nhau.

|  |  |
| --- | --- |
| **Function** | **Description** |
| [attr()](https://www.w3schools.com/cssref/func_attr.asp) | Returns the value of an attribute of the selected element |
| [calc()](https://www.w3schools.com/cssref/func_calc.asp) | Allows you to perform calculations to determine CSS property values |
| [cubic-bezier()](https://www.w3schools.com/cssref/func_cubic-bezier.asp) | Defines a Cubic Bezier curve |
| [hsl()](https://www.w3schools.com/cssref/func_hsl.asp) | Defines colors using the Hue-Saturation-Lightness model (HSL) |
| [hsla()](https://www.w3schools.com/cssref/func_hsla.asp) | Defines colors using the Hue-Saturation-Lightness-Alpha model (HSLA) |
| [linear-gradient()](https://www.w3schools.com/cssref/func_linear-gradient.asp) | Sets a linear gradient as the background image. Define at least two colors (top to bottom) |

THAM CHIẾU BỘ CHỌN CSS

## Bộ chọn CSS

Trong CSS, bộ chọn là các mẫu được sử dụng để chọn (các) phần tử bạn muốn tạo kiểu.

Sử dụng Trình [kiểm tra bộ chọn CSS](https://www.w3schools.com/cssref/trysel.asp) của chúng tôi để trình bày các bộ chọn khác nhau.